sun microsystems

SPARCserver Manager User Guide

Sun Microsystems, Inc. • 2550 Garcia Avenue • Mountain View, CA 94043 • 415-960-1300

Part No: 800-3838-10
Revision A of 29 June 1990

The Sun logo, Sun Microsystems, Sun Workstation, NFS, and TOPS are registered trademarks of Sun Microsystems, Inc.

Sun, Sun-2, Sun-3, Sun-4, Sun386*i*, SPARCstation, SPARCserver, NeWS, NSE, OpenWindows, SPARC, SunInstall, SunLink, SunNet, SunOS, SunPro, and SunView are trademarks of Sun Microsystems, Inc.

UNIX is a registered trademark of AT&T; OPEN LOOK is a trademark of AT&T.

All other products or services mentioned in this document are identified by the trademarks or service marks of their respective companies or organizations, and Sun Microsystems, Inc. disclaims any responsibility for specifying which marks are owned by which companies or organizations.

SPARCserver Manager, MetaDisk, MetaDisk Driver, metapartition, and IOadmin are trademarks of Sun Microsystems, Inc.

Copyright © 1990 Sun Microsystems, Inc. – Printed in U.S.A.

All rights reserved. No part of this work covered by copyright hereon may be reproduced in any form or by any means – graphic, electronic, or mechanical – including photocopying, recording, taping, or storage in an information retrieval system, without the prior written permission of the copyright owner.

Restricted rights legend: use, duplication, or disclosure by the U.S. government is subject to restrictions set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and in similar clauses in the FAR and NASA FAR Supplement.

The Sun Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees.

This product is protected by one or more of the following U.S. patents: 4,777,485 4,688,190 4,527,232 4,745,407 4,679,014 4,435,792 4,719,569 4,550,368 in addition to foreign patents and applications pending.

This software and documentation is based in part on the Fourth Berkeley Software Distribution under license from the Regents of the University of California. We acknowledge the following individuals and institutions for their role in its development: The Regents of the University of California, the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California, and Other Contributors.

Welcome to SPARCserver Manager

SPARCserver Manager™ is a software package that includes the MetaDisk Driver™ and a suite of three software tools that simplify the administration and management of Sun systems. As a system administrator you can use these tools (collectively called IOadmin™) to look at network servers, spot problems, and take corrective action. The functions of MetaDisk™ and the three IOadmin tools are briefly outlined here, and are described in detail in subsequent chapters.

- **MetaDisk Driver** — used to create large *metapartitions*™ (up to two gigabytes), which can be manipulated just as standard disk partitions are; MetaDisk also allows for *mirroring* of disk partitions, thus providing protection from a single-disk failure and online backups with no downtime.
- **Device Management** (`dmt`) — information about and management of hardware devices, including servers, controllers, disk drives, and tape drives.
- **File System Management** (`fsm`) — information about and management of files and file systems, including creating and checking file systems and displaying information about individual directories and files.

  File System Management includes access to and manipulation of MetaDisks and associated metapartitions.
- **Performance Monitor** (`pmt`) — information about the current or historical performance of a server; most of the information is in graph form.

Read this document tc learn how to install and use MetaDisk and IOadmin. For more information about related subjects, see also:

- *SunView User's Guide* (information about SunView windows)
- *System and Network Administration Guide* (information about networks and kernel configuration)

Contents

Welcome to SPARCserver Manager .....

Chapter 1 Introduction **1**

1.1. Document Organization 1

1.2. Definition of Terms 2

1.3. Typographic Conventions 2

Chapter 2 Getting Started **5**

2.1. Hardware and Software Requirements 5

Common Requirements 5

MetaDisk Driver Requirements 5

IOadmin Requirements 6

2.2. Local Installation 6

Disk Space Requirements 7

2.3. Remote Installation 9

Disk Space Requirements 10

2.4. IOadmin Post-Installation Procedures 12

Start the RPC daemon on Servers 12

Include daemon Startups in rc Files on Servers 13

Start dmt Logging During Reboot of Server 14

Set command paths for running IOadmin on IO Manager 14

Move man Pages 14

Mount the IOadmin Tools 15

2.5. Building a MetaDisk Driver Kernel 15

Files Affected by MetaDisk Driver Installation 15

SunOS 4.0.3 System 15
SunOS 4.1 System 15
Making a GENERIC MetaDisk Kernel 16
Kernel Must Be /vmunix 17
Making a Customized Kernel 17
Eliminating Non-Existent Devices 17
Building and Booting the New Kernel 17
Kernel Must Be /vmunix 18
2.6. MetaDisk Post-Installation Procedures for Servers 18
Include MetaDisk Driver Commands in root's Path 19
Moving MetaDisk Driver man Pages 19
Starting the Metadisk Driver Daemon During Reboot 19
Activating Metapartitions During Reboot 19
Resyncing Mirrored Metapartitions During Reboot 20
Mounting Metapartitions During Reboot 21

Chapter 3 MetaDisk Driver Basics **23**
3.1. When to Use MetaDisk 24
3.2. Overview 24
Definition of Terms 24
MetaDisk Commands 25
3.3. Planning Metapartitions 28
Performance and Reliability Ramifications of Using MetaDisk 29
Naming Metapartitions 29
Raw Device Names 30
Partitions a, b, c Have No Special Meaning 30
3.4. Defining Metapartitions: the metatab File 30
Sample metatab File 31
3.5. Activating Metapartitions with metainit 32
Using fsm to Prevent Problems 32
How You Can Use Metapartitions 33
Activating All Metapartitions at One Time 33
Activating One Metapartition at a Time 34

Error Conditions 34
Non-Existent Metapartition 34
Non-Existent Physical Partition 35
Physical Partition Used Twice 35
Metapartition Name Used Twice 35
Not Enough Rows Defined 35
Activating Metamirror Before Metapartitions 36
3.6. Checking the Status of Metapartitions 36
3.7. Clearing Metapartitions 38
Clearing a Single Metapartition 38

Chapter 4 MetaDisk Driver Tutorial **39**
4.1. Summary: Installing and Using Metadisk Driver 39
Installing Software and Building Kernel 39
Defining and Activating Metapartitions 39
Making and Using File Sytems on Metapartitions and Metamirrors 39
4.2. Cautions and Admonitions 40
4.3. Defining Metapartitions 45
Defining a Simple Metapartition 45
One Physical Partition per Metapartition 45
Multiple Physical Partitions (one disk) 45
Multiple Physical Partitions (two full disks) 46
Creating metatab Entry with metastat 46
Defining a Metamirror 46
Metamirror: One Metapartition Defined 46
Metamirror: Two Metapartitions, One Physical Partition per Metapartition 47
Metamirror: Multiple Physical Partitions in Metapartition 47
Metamirror: -r and -s Options 47
Metamirror: -g Option 47
Metamirror: Read-Error Threshold (-e) and Pass Number Options 48
Sample Metatab 48

4.4. Activating, Checking, and Clearing Metapartitions 48
Activating Metapartitions 49
Activating a Single Metapartition 49
Activating a Single Metamirror 49
Checking Metapartition Status 49
Clearing Metapartitions 49
4.5. Modifying an Unused Metapartition or Metamirror 50
If the Metapartition Has Not Been Activated 50
If the Metapartition Has Been Activated 50
Adding a Second Metapartition to a Metamirror 50
Making a Physical Partition into a Metamirror Component 51
Example 51
4.6. Using Metapartitions and Metamirrors 53
Making and Mounting a File System 53
Syncing Mirrored Metapartitions 54
Watching Progress of Resync 55
Performing Online Backups — Mirroring Active 55
Example 56
Performing Online Backups — Mirroring Not Active 57
Example 57
4.7. Modifying a Used Metapartition 59
Changing the Definition of a Used Metapartition 59
Replacing a Disk: Concatenated Metapartition 60
Mirroring: Replacing a Failed Disk Partition — Spare Disk Available 61
Mirroring: Replacing a Failed Disk Partition — No Spare Disk Available 62
4.8. Rebooting Without Resyncing 63
4.9. Safety Tips 63
4.10. How the MetaDisk Handles Errors 64
4.11. Performance Tips 65

Chapter 5 IOadmin Overview **67**

5.1. About IOadmin Directories 67
5.2. Windows: The SunView Environment 67
Primary IOadmin Tool Windows 68
Popup Windows 69
Text Subwindows 69
Dismissing Popup Windows 69
Scrollable Canvases and Subwindows 69
Moving a Window 70
Resizing a Window 70
Printing a SunView Screen 70
5.3. Buttons and Menus 70
Mouse Buttons 70
View and Do Menus 71
Do Menu: Need for root Password 71
Server Name Menu 71
Cycle Buttons 72
5.4. Starting IOadmin Tools 72
Command Line Options 73
Starting All Three Tools at One Time 73
Specifying Servers to Be Monitored 73
5.5. Including Devices in ioadmd.conf 74
Contents of `ioadmd.conf` 74
Preventing Unwanted Side Effects 74
Speeding Up fsm 75
Why fsm Is Slowed Down 75
Preventing Undue Delays 75
How ioadmd.conf Works 76
Sample ioadmd.conf File 77
5.6. Changing Servers 77
5.7. Tool Icons: Closing and Opening 79
Closing a Tool to an Icon 79
Opening a Tool from an Icon 79
5.8. Quitting IOadmin Tools 79

5.9. Log Files 80

Chapter 6 Device Management **81**

6.1. Starting Device Management 81

The Device Management Window 82

6.2. Closing or Quitting Device Management 82

Closing Device Management to an Icon 82

Quitting Device Management 82

6.3. The System Map 82

Displaying the System Map 82

Selecting a Device in the System Map 83

Dynamic Hardware Status Indicators 84

6.4. The View Menu 84

Popup Windows 85

Informational Popups 85

Note Pad Popup 86

Error Log Popups 86

Displaying an Error Log 87

Clearing Panic Error Flag 87

6.5. The Do Menu 87

Updating the Display 88

Starting dmt Logging 88

Running Disk Test 89

Running File Test 90

Chapter 7 File System Management **93**

7.1. Terminology 93

7.2. Starting File System Management 94

7.3. Closing or Quitting File System Management 95

Closing File System Management to an Icon 95

Quitting File System Management 95

7.4. The View Menu 95

Disk Layout 95

Disk Layout — Metapartition Selected 96
Partitions 97
Partitions - All but "c" 97
Partitions - with File Systems 98
Partitions - without File Systems 98
Partitions - Available 99
Partitions - All 99
7.5. Selecting a Partition 99
Selecting Multiple Partitions 99
7.6. Selection Detail 100
Physical Partition 100
Metapartition Detail 100
7.7. File System Usage 101
7.8. File Information 102
7.9. The Do Menu 102
Update Display 102
Create File System 103
Check File System 103
Mount File System 104
Unmount File System 105
Metapartitions 105
Activating, Clearing, or Resyncing Metapartitions 105

Chapter 8 Performance Monitor **107**
8.1. Starting Performance Monitor 107
8.2. Closing or Quitting Performance Monitor 108
Closing Performance Monitor to an Icon 108
Quitting Performance Monitor 108
8.3. The View Menu 108
Logged or Current Data? 109
Bar or Line Graph? 109
Grid On or Off? 109
Tabular Data? 109

8.4. Viewing Current Data 110
8.5. Logging 111
Start Logging 111
Stop Logging 112
Change Logging Interval 112
8.6. Disk Storage Considerations 112
8.7. Viewing Logged Data 112
For a Time Period 112
Load a Specific File 113
CPU Utilization - Bar Graph 114
CPU Utilization - Line Graph 115
APPENDIX A: Man Pages **117**

Index **119**

Figures

Figure 3-1 MetaDisk Driver Position in Kernel 23
Figure 3-2 FSM: Disk Layout 28
Figure 3-3 Concatenated Metapartition 33

Figure 4-1 Mirrored Metapartitions 42
Figure 4-2 FSM: Unmounted File System 45

Figure 5-1 Elements of an IOadmin Window 68
Figure 5-2 Server Name Menu 72
Figure 5-3 Server Name Menu: Server Selected 78
Figure 5-4 IOadmin Tools Icons 79

Figure 6-1 Device Management Icon 82
Figure 6-2 System Map 83
Figure 6-3 System Map: Server Selected 84
Figure 6-4 View Menu: Server Selected 85
Figure 6-5 View Errors Menu 87
Figure 6-6 Do Menu 88
Figure 6-7 Disk Test Popup Window 89
Figure 6-8 File Test 91

Figure 7-1 File System Management Icon 95
Figure 7-2 FSM: Disk Layout 96
Figure 7-3 FSM: Disk Layout - Metapartition Selected 96
Figure 7-4 FSM: Partition Menu - All but ''c'' Selected 97

Figure 7-5 FSM: Partitions With File Systems 98
Figure 7-6 Partitions - Available 99
Figure 7-7 FSM: Selection Detail - xy0d 100
Figure 7-8 Selection Detail - md0c 100
Figure 7-9 File System Usage 101
Figure 7-10 File Information 102
Figure 7-11 Create File System Selected 103
Figure 7-12 File System Check Warning 104
Figure 7-13 File System Check 104
Figure 7-14 Do Menu: Metapartition Resync 106

Figure 8-1 Performance Monitor Icon 108
Figure 8-2 View Menu 109
Figure 8-3 Graph Data 110
Figure 8-4 Paging Activity 111
Figure 8-5 Logging Status Window 111
Figure 8-6 Logged Data Window 113
Figure 8-7 "Load For a Time Period" Selected 113
Figure 8-8 Load A Specific File 114
Figure 8-9 CPU Utilization: Bar Graph 115
Figure 8-10 CPU Utilization: Line Graph 115

1

Introduction

1.1. Document Organization

This document is comprised of this introduction and seven other chapters: getting started (including installation of the MetaDisk Driver and IOadmin), MetaDisk Driver basics, MetaDisk Driver tutorial, IOadmin overview, and one for each for the IOadmin tools. Appendix A is a copy of the `man` pages, which are also on the installation tape.

- Chapter 1, "Introduction," outlines what is in this document, describes the typographic conventions used within it, and defines some general terminology.
- Chapter 2, "Getting Started," tells you what you need to do before you can use SPARCserver Manager, including installation of the MetaDisk Driver and IOadmin. This chapter also tells you how to configure a kernel that recognizes the MetaDisk Driver.
- Chapter 3, "MetaDisk Driver Basics," describes a new concept — *metapartitions* — and briefly explains how to create and use them.
- Chapter 4, "MetaDisk Driver Tutorial," provides many examples of metapartitions and explains their use in detail, including cautionary rules.
- Chapter 5, "IOadmin Overview," defines some terms and explains how they are used in IOadmin, and discusses the layout and logic of the windows, icons, and menus in IOadmin.
- Chapter 6, "Device Management," describes `dmt`, the tool used to monitor and manage the hardware devices that make up the systems you administer.
- Chapter 7, "File System Management," describes `fsm` and how to use it to view, create, mount, unmount, and check files and file systems; includes information about and functions to be performed on metapartitions.
- Chapter 8, "Performance Monitor," explains `pmt`, the IOadmin tool that graphically monitors the performance of each of your servers.
- Appendix A, "Man Pages," is a hard copy of the man pages, which are included on the installation tape.

1.2. Definition of Terms

The following definitions are general definitions for SPARCserver Manager; see Chapter 3 for specific MetaDisk Driver definitions and Chapter 5 for specific IOadmin definitions.

- **MetaDisk Driver:** Software that defines the MetaDisk and allows you to create *virtual disks*, or *metapartitions*. See Chapter 3 for more MetaDisk Driver definitions.
- **IOadmin:** The set of three *tools* that help you monitor and administer devices, file systems, and performance. See Chapter 5 for more IOadmin definitions.
- **IO Manager:** A machine from which you use IOadmin to monitor and administer other machines, called *IO servers*.
- **IO Server:** Any machine which is monitored and administered by IOadmin, usually through an *IO Manager*; also called simply a *server*.
- **Tool:** Software that does a particular job. The three IOadmin *tools* are Device Management, File System Management, and Performance Monitor.
- **Package:** A group of *tools* which serve a unified purpose. The SPARCserver Manager *package* helps you administer and manage your Sun systems.

1.3. Typographic Conventions

The following typographic conventions are used in this manual:

- **bold font**

  The **bold font** is used in step-by-step, numbered instructions, to indicate what you should do. An explanation of the step may follow in a normal font.

  1. **Do this.**

     Read this explanation.

  2. **Then do this.**

  The **bold font** is also used in glossary-like explanations of terms to make the term being described stand out.

- `listing font`

  This `font` is used to indicate system responses displayed on the screen, for literal values such as program, function, procedure, or variable names, and for listings of the contents of a file.

- [keyboard] buttons

  A multi-character entry within a box (such as [Return] or [L1], represents pressing a keyboard button.

- boxed entries

  Text within a box represents screen displays or the listing of the contents of a file. There is no user interaction shown in such displays.

```
Sep 20 11:24:57 brahms vmunix: Accounting suspended
Sep 20 11:24:59 brahms vmunix: /: file system full
Sep 20 11:24:59 brahms vmunix: Accounting resumed
```

- **bold listing font**

  This **font** represents text that you type.

- Gray boxes

  Material within a gray box represents a dialog you have with the system: in response to a system prompt or other activity, you type in something. The **bold** text in a gray box represents your response.

```
Enter name of IO Manager machine:
opusone
Enter name of server:
fourfox
```

- *italic font*

  The *italic font* represents a variable for which you need to substitute the proper value, as in the example below; this font is also used for *emphasis* or to introduce a *new term* into the text.

```
# cd /sys/sun?/conf
```

- **Exploding Workstation**

  Marks information that it is *imperative* for you to read and pay attention to. If you ignore this material, you may damage your system or jeopardize the installation.

- **Pointing Hand**

  Points to information that requires close attention.

2

Getting Started

SPARCserver Manager includes the MetaDisk Driver and three IOadmin tools: Device Management, File System Management, and Performance Monitor. Follow the instructions in this chapter to install the IOadmin tools and the MetaDisk Driver, and to configure a MetaDisk Driver kernel.

2.1. Hardware and Software Requirements

Note the following requirements for installing and running the MetaDisk Driver and IOadmin. The requirements are divided into three categories:

- Requirements for either MetaDisk Driver or IOadmin
- Requirements for MetaDisk Driver only
- Requirements for IOadmin only

If MetaDisk Driver or IOadmin do not work after they have been installed, check that each of the requirements below has been met.

Common Requirements

To run MetaDisk Driver or IOadmin, the following conditions must be met:

- Each licensed server must have at least one attached hard disk.
- Each licensed server and each IO manager must be running SunOS 4.0.3 or SunOS 4.1.
- Each licensed server and each IO manager must be a Sun-4 (Sun-4c OK for IOadmin only).

MetaDisk Driver Requirements

For the MetaDisk Driver, the following conditions must also be met, in addition to the common conditions listed for MetaDisk Driver and IOadmin above:

- The MetaDisk utility programs must be installed on each licensed server; you will need approximately 850 kilobytes of disk space in `/usr/etc` and 120 kilobytes in `/sys`.
- The MetaDisk Driver must be configured into the kernel for each licensed server, and the server must be rebooted with the new kernel.
- The file `/etc/metatab`, describing the metapartitions to be used, must exist (see Chapter 3).

IOadmin Requirements

For IOadmin, the following conditions must also be met, in addition to the common conditions listed for MetaDisk Driver and IOadmin, above:

- The IOadmin RPC daemon (`rpc.ioadmd`) must be running on each licensed server.
- The IOadmin daemon directory (`ioadm/etc`) must be installed on each licensed server.
- The IOadmin tools directory (`ioadm/bin`, where `dmt`, `fsm`, and `pmt` reside) must be installed (or mounted) on each IO manager; you will need approximately 750 kilobytes of disk space for IOadmin.
- Any machine on which you run the IOadmin tools must be running SunView. (Note, however, that you can monitor a server remotely, from an IO Manager running SunView, even if the server is not running SunView.)
- The IO manager must have direct RPC access to the IO servers; that is, the servers must be accessible through a network.

2.2. Local Installation

Install IOadmin and the MetaDisk Driver on each licensed server; install IOadmin on each IO Manager. Do the following if you have a 1/4" cartridge tape drive on the machine on which you are performing the installation. If you wish to perform a remote installation, see "Remote Installation" below.

Install the MetaDisk Driver and the IOadmin directories as described below. Note that the MetaDisk Driver and the IOadmin daemon directory can only be installed on licensed servers; the IOadmin tools directory can be installed on any machine.

The following installations should be performed on one or more machines, depending on how many licenses you have purchased.

Step 1: Become root

You will need to be root to create some of the directories and files needed for the installation.

Step 2: Kill IOadmin daemons, if necessary

If you have previously installed IOadmin, make sure that `pmlogd` and `rpc.ioadmd` are not running. Kill the processes if they are running. Substitute the actual process ID number for 123 and 3586 in the following example:

```
# ps aux | grep rpc.ioadmd
root  3586  0.0  2.6  104  88 ?  I  May 24  5:13 rpc.ioadmd
# kill 3586
ps aux | grep pmlogd
root  123  0.0  0.0  88  0 ?  IW  11:23  0:00
/usr/ioadm/etc/pmlogd
# kill 123
```

Step 3: Mount SPARCserver Manager installation tape

Load the SPARCserver Manager installation tape into your tape drive. The tape includes SunOS 4.0.3 and SunOS 4.1 files; the script figures out which OS you are running and uses the correct files. The physical label on the tape should read as follows:

```
1.0 SPARCserver Manager™ Sun-4™
1/4" Tape (tar format, QIC24, 1 of 1
Part Number:  700-2592-10 Rev. 01
```

Step 4: Run installation script

IOadmin and MetaDisk Driver are installed by way of the UNIX utility, `extract_unbundled`. As root, type

```
# /usr/etc/extract_unbundled
```

and respond to the prompts, as appropriate. In the following example, a **local** installation of IOadmin and MetaDisk Driver is performed, from tape device **/dev/rst8**; IOadmin is installed in **/home/new/ioadm** instead of the default location, `/usr/ioadm`.

Note the names and sizes of the files on the tape; the same files should be installed when you run `extract_unbundled`. The sizes may vary slightly.

Disk Space Requirements

As noted in the `extract_unbundled` script, IOadmin requires about 750 kilobytes of disk space. The MetaDisk Driver requires about 850 kilobytes in `/usr/etc` and about 120 kilobytes in `/sys`.

```
# /usr/etc/extract_unbundled                          <==== (what you type = bold)
Enter media drive location [local | remote]: local    <==== (what you type = bold)
Enter Device Name (e.g. rst0, rmt0, rfd0c) : /dev/rst8  <==== (what you type = bold)
**Please mount the release media if you haven't done so already.**
Press return when ready:Return
The following product will be installed:
3+0 records in
3+0 records out
    1.0 SPARCserver Manager(TM)
    Sun-4 SUNBIN
    1/4" Tape (tar format, QIC 24 ) 1 of 1
    Part Number:  700-2592-10
    Compatible with the SunOS 4.0.3 and SunOS 4.1 releases
   [Copyright information and Restricted Rights Legend]
    (continued on next page)
```

```
    (continued from previous page)
Do you want to continue [y|n]? y                    <==== (what you type = bold)
/usr/etc/extract_unbundled : Extracting Installation Scripts
tar: blocksize = 20
x ./install_unbundled, 5764 bytes, 12 tape blocks

/usr/etc/extract_unbundled : Begin Install Script Execution

SPARCserver Manager installation script

Installation of IOadmin requires approximately 750 Kbytes of disk space.

Installation of MetaDisk requires approximately 850 Kbytes of disk
space in /usr/etc and approximately 120 Kbytes of disk space in /sys.

Install IOadmin tools for Sun 4 [y|n]? y            <==== (what you type = bold)

Install IOadmin in /usr/ioadm [y|n]? n              <==== (what you type = bold)

Install IOadmin where? /home/new/ioadm              <==== (what you type = bold)

Install MetaDisk for Sun 4 [y|n]? y                 <==== (what you type = bold)

creating /home/new/ioadm for the IOadmin tools...

Setting current directory to /home/new/ioadm

Positioning the tape...
Installing the IOadmin tools...
x ./etc/rpc.ioadmd, 73728 bytes, 144 tape blocks
x ./etc/syslog_conf.sed, 362 bytes, 1 tape blocks
x ./etc/disktest, 40960 bytes, 80 tape blocks
x ./etc/filetest, 32768 bytes, 64 tape blocks
x ./etc/pmlogd, 40960 bytes, 80 tape blocks
x ./man/fsm.1, 7003 bytes, 14 tape blocks
x ./man/pmlogd.8, 1801 bytes, 4 tape blocks
x ./man/ioadmd.8C, 1047 bytes, 3 tape blocks
x ./man/pmt.1, 3190 bytes, 7 tape blocks
x ./man/dmt.1, 7106 bytes, 14 tape blocks
x ./bin/dmt, 172032 bytes, 336 tape blocks
x ./bin/pmt, 163840 bytes, 320 tape blocks
x ./bin/fsm, 163840 bytes, 320 tape blocks

changing directory to /usr/etc

Positioning the tape...
Installing the MetaDisk Driver utilities...
x ./metainit, 90112 bytes, 176 tape blocks
x ./metaclear, 81920 bytes, 160 tape blocks
x ./metastat, 73728 bytes, 144 tape blocks
x ./metasync, 81920 bytes, 160 tape blocks
x ./metareset, 81920 bytes, 160 tape blocks
x ./metaoffline, 81920 bytes, 160 tape blocks
x ./metad, 81920 bytes, 160 tape blocks
x ./metareplace, 81920 bytes, 160 tape blocks
x ./metattach, 81920 bytes, 160 tape blocks

    (continued on next page)
```

```
    (continued from previous page)
x ./metadetach, 81920 bytes, 160 tape blocks
x ./metafasthalt, 151 bytes, 1 tape blocks
x ./metafastboot, 155 bytes, 1 tape blocks
x ./metahalt, 124 bytes, 1 tape blocks
x ./metareboot, 130 bytes, 1 tape blocks

changing directory to /sys/sun

Positioning the tape...
Installing the MetaDisk Driver files and binaries...
x ./buf.h.md, 7370 bytes, 15 tape blocks
x ./mdreg.h, 10466 bytes, 21 tape blocks
x ./md.o, 34170 bytes, 67 tape blocks
x ./md_subr.o, 8113 bytes, 16 tape blocks
x ./md.41.o, 34224 bytes, 67 tape blocks
x ./md_subr.41.o, 8288 bytes, 17 tape blocks
x ./vfs_bio.o.md, 9543 bytes, 19 tape blocks
x ./ufs_vfsops.o.md, 6837 bytes, 14 tape blocks
x ./metascript, 12696 bytes, 25 tape blocks
x ./metatab.5, 7091 bytes, 14 tape blocks
x ./metautil.8, 8690 bytes, 17 tape blocks
x ./make_md, 1233 bytes, 3 tape blocks
#
```

Go to Section 2.4, "IOadmin Post-Installation Procedures" to continue preparing to use IOadmin and the MetaDisk Driver.

2.3. Remote Installation

Install IOadmin and the MetaDisk Driver on each licensed server; install IOadmin on each IO Manager. Do the following if you do not have a 1/4" cartridge tape on the machine on which you are performing the installation. If you wish to perform a local installation, see "Local Installation" above.

Install the MetaDisk Driver and the IOadmin directories as described below. Note that the MetaDisk Driver and the IOadmin daemon directory can only be installed on licensed servers; the IOadmin tools directory can be installed on any machine.

The following installations should be performed on one or more machines, depending on how many licenses you have purchased.

Step 1: Become root on installation machine

You will need to be root on the machine on which you will be doing the installation — the machine without a 1/4" cartridge tape drive.

Step 2: Kill IOadmin daemons on installation machine, if necessary

If you have previously installed IOadmin on the machine on which you will be doing the installation, make sure that `pmlogd` and `rpc.ioadmd` are not running. Kill the processes if they are running. Substitute the actual process ID number for 123 and 3586 in the following example:

```
# ps aux | grep rpc.ioadmd
root  3586  0.0  2.6  104  88 ?  I  May 24  5:13 rpc.ioadmd
# kill 3586
ps aux | grep pmlogd
root  123  0.0  0.0  88  0 ?  IW  11:23  0:00
/usr/ioadm/etc/pmlogd
# kill 123
```

Step 3: Edit `/.rhosts` on tapehost

As root on the *tapehost* (the machine with the 1/4" cartridge tape drive), edit `/.rhosts`, inserting the hostname of the machine on which you will be doing the installation (the machine with no 1/4" cartridge tape drive):

Step 4: Mount SPARCserver Manager installation tape on tapehost

Load the SPARCserver Manager installation tape into the tape drive on the tapehost, the machine from which you will perform the installation.

Step 5: Run installation script on installation machine

IOadmin and MetaDisk Driver are installed by way of the UNIX utility, `extract_unbundled`. As root, type the following on the machine on which you will be doing the installation:

```
# /usr/etc/extract_unbundled
```

and respond to the prompts, as appropriate. In the following example, a **remote** installation of IOadmin and MetaDisk Driver is performed, from tape device **`/dev/rst8`**; IOadmin is installed in the default directory, `/usr/ioadm`.

Note the names and sizes of the files on the tape; the same files should be installed when you run `extract_unbundled`. The sizes may vary slightly.

Disk Space Requirements

As noted in the `extract_unbundled` script, IOadmin requires about 750 kilobytes of disk space. The MetaDisk Driver requires about 850 kilobytes in `/usr/etc` and about 120 kilobytes in `/sys`.

```
# /usr/etc/extract_unbundled                          <==== (what you type = bold)
Enter media drive location [local | remote]: remote   <==== (what you type = bold)
Enter hostname of remote drive: buenos                <==== (what you type = bold)
Enter Device Name (e.g. rst0, rmt0, rfd0c, rfd0g) : /dev/rst8   <==(your entry=bold)
**Please mount the release media if you haven't done so already.**
Press return when ready:Return
The following product will be installed:
3+0 records in
3+0 records out
   [Tape label, Copyright information, and Restricted Rights Legend]
       (continued on next page)
```

```
        (continued from previous page)

Do you want to continue [y|n]? y                          <==== (what you type = bold)
/usr/etc/extract_unbundled : Extracting Installation Scripts
0+1 records in
0+1 records out
tar: blocksize = 20
x ./install_unbundled, 5764 bytes, 12 tape blocks

/usr/etc/extract_unbundled : Begin Install Script Execution

SPARCserver Manager installation script (beta)

Installation of IOadmin requires approximately 750 Kbytes of disk space.

Installation of MetaDisk requires approximately 850 Kbytes of disk
space in /usr/etc and approximately 120 Kbytes of disk space in /sys.

Install IOadmin tools for Sun 4 [y|n]? y                  <==== (what you type = bold)

Install IOadmin in /usr/ioadm [y|n]? y                    <==== (what you type = bold)

Install MetaDisk for Sun 4 [y|n]? y                       <==== (what you type = bold)

Setting current directory to /usr/ioadm
Positioning the tape...

Installing the IOadmin tools...
x ./etc/rpc.ioadmd, 73728 bytes, 144 tape blocks
x ./etc/syslog_conf.sed, 362 bytes, 1 tape blocks
x ./etc/disktest, 40960 bytes, 80 tape blocks
x ./etc/filetest, 32768 bytes, 64 tape blocks
x ./etc/pmlogd, 40960 bytes, 80 tape blocks
x ./man/fsm.1, 7003 bytes, 14 tape blocks
x ./man/pmlogd.8, 1801 bytes, 4 tape blocks
x ./man/ioadmd.8C, 1047 bytes, 3 tape blocks
x ./man/pmt.1, 3190 bytes, 7 tape blocks
x ./man/dmt.1, 7106 bytes, 14 tape blocks
x ./bin/dmt, 172032 bytes, 336 tape blocks
x ./bin/pmt, 163840 bytes, 320 tape blocks
x ./bin/fsm, 163840 bytes, 320 tape blocks
11+1 records in
11+1 records out

changing directory to /usr/etc

Positioning the tape...
Installing the MetaDisk Driver utilities...
x ./metainit, 90112 bytes, 176 tape blocks
x ./metaclear, 81920 bytes, 160 tape blocks
x ./metastat, 73728 bytes, 144 tape blocks
x ./metasync, 81920 bytes, 160 tape blocks
x ./metareset, 81920 bytes, 160 tape blocks
x ./metaoffline, 81920 bytes, 160 tape blocks
x ./metad, 81920 bytes, 160 tape blocks

        (continued on next page)
```

```
        (continued from previous page)

x ./metareplace, 81920 bytes, 160 tape blocks
x ./metattach, 81920 bytes, 160 tape blocks
x ./metadetach, 81920 bytes, 160 tape blocks
x ./metafasthalt, 151 bytes, 1 tape blocks
x ./metafastboot, 155 bytes, 1 tape blocks
x ./metahalt, 124 bytes, 1 tape blocks
x ./metareboot, 130 bytes, 1 tape blocks

changing directory to /sys/sun

Positioning the tape...
Installing the MetaDisk Driver files and binaries...
x ./buf.h.md, 7370 bytes, 15 tape blocks
x ./mdreg.h, 10466 bytes, 21 tape blocks
x ./md.o, 34170 bytes, 67 tape blocks
x ./md_subr.o, 8113 bytes, 16 tape blocks
x ./md.41.o, 34224 bytes, 67 tape blocks
x ./md_subr.41.o, 8288 bytes, 17 tape blocks
x ./vfs_bio.o.md, 9543 bytes, 19 tape blocks
x ./ufs_vfsops.o.md, 6837 bytes, 14 tape blocks
x ./metascript, 12696 bytes, 25 tape blocks
2+1 records in
2+1 records out
x ./metatab.5, 7091 bytes, 14 tape blocks
x ./metautil.8, 8690 bytes, 17 tape blocks
x ./make_md, 1233 bytes, 3 tape blocks
#
```

Step 6: Edit `/.rhosts` on tapehost

Remove the name of the installation machine from `/.rhosts` on the tapehost, the machine from which you did the installation.

2.4. IOadmin Post-Installation Procedures

To continue the preparation for using IOadmin, you must start the IOadmin RPC daemon on each server to be monitored. You may also wish to include IOadmin daemon startups in `rc.local` on each server and set command paths for running IOadmin on your IO Manager.

Other possible post-installation procedures include moving the IOadmin man pages and mounting the IOadmin tools instead of installing them on every IO Manager machine. Each of these procedures is described below.

Start the RPC daemon on Servers

As root, start `rpc.ioadmd` running on each server on which have installed IOadmin (you will have to be in the correct IOadmin directory or include the full path name unless you have included the IOadmin directory in the command path for root):

```
# rpc.ioadmd
```

When you start the daemon, `rpc.ioadmd` does the following:

- looks for the `ioadm` spool directory, creating it if it does not exist:

```
/var/spool/ioadm
```

- creates a lock file (`/var/spool/ioadm/ioadmd.lock`), which prevents multiple instances of `rpc.ioadmd` from running.
- looks for the `ioadm` logging directories, creating them if they do not exist:

```
/var/spool/ioadm/pmlog
/var/spool/ioadm/dmtlog
```

- creates a `syslog.conf` file and empty log files in `dmtlog`

Note that the daemon, `rpc.ioadmd`, expects the following files to be in the same directory as the daemon, which they are after installation:

```
disktest
filetest
pmlogd
rpc.ioadmd
syslog_conf.sed
```

Include daemon Startups in rc Files on Servers

Include starting of `rpc.ioadmd` in `/etc/rc.local` on each server so that the daemon will always be restarted after a system reboot. You may also want to include `pmlogd` in `rc.local`; this will automatically start Performance Monitor logging on the server. Otherwise you will have to start Performance Monitor logging manually.

If you have installed IOadmin somewhere other than the default location, substitute the correct path for `/usr/ioadm` in the following example.

```
 . . .
#
# start up status monitor and locking daemon if present
#
if [ -f /usr/etc/rpc.statd ]; then
      rpc.statd &              (echo -n ' statd')      >/dev/console
fi
if [ -f /usr/etc/rpc.lockd ]; then
      rpc.lockd &              (echo -n ' lockd')      >/dev/console
fi
if [ -f /usr/ioadm/etc/rpc.ioadmd ]; then
      /usr/ioadm/etc/rpc.ioadmd &      (echo -n ' ioadmd')      >/dev/console
fi

if [ -f /usr/ioadm/etc/pmlogd ] ; then
      /usr/ioadm/etc/pmlogd&           (echo -n 'pmlogd')       >/dev/console
fi

#
# start up authentication daemon if present and if adjunct file exists
 . . .
```

Start dmt Logging During Reboot of Server

To cause `dmt` logging instead of the standard `syslogd` to be started automatically during bootup of your server, modify the `syslogd` entry in `/etc/rc.local`, adding the `-f` argument shown in bold below:

```
 . . .
# syslogd doesn't belong here, but needs to be started before the others.
if [ -f /usr/etc/syslogd ]; then
                     (echo 'starting system logger') >/dev/console
      rm -f /dev/log
        syslogd -f/var/spool/ioadm/dmtlog/syslog.conf
fi
 . . .
```

Set command paths for running IOadmin on IO Manager

You may want to include the IOadmin tools directory (by default, it is `/usr/ioadm/bin`) in your path variable on the machine or machines on which you will run IOadmin.

Move man Pages

The man pages for IOadmin and MetaDisk Driver are installed on your system when you run the `extract_unbundled` script. IOadmin man pages are installed wherever you install IOadmin (`/usr/ioadm/man` by default). You might wish to copy or move the man pages to some central man pages location for easy access.

Mount the IOadmin Tools

Although it is recommended that you install the IOadmin tools on each IO Manager (in `/usr/ioadm/bin`, by default), you could install the tools on one licensed server and mount them from that server. If you do mount the tools, **hard-mount** them only; if they are soft-mounted, they may crash if you experience network interruptions or timeouts.

2.5. Building a MetaDisk Driver Kernel

Do one of the following procedures to configure and build a kernel that recognizes the MetaDisk Driver; you can modify the GENERIC kernel or your own custom kernel, adding one line for MetaDisk Driver support. Note that the MetaDisk Driver will run only on Sun-4 machines (not on Sun-4c).

As root, run `metascript` to copy and modify the necessary files:

```
# cd /sys/sun
# metascript
Checking /sys/sun/conf.c...                    (find major device number; add data structure to conf.c)
Major numbers used: 4, 0                       (major number for raw device = 4; for block device = 0)
Making MetaDisk devices...                     (making devices in /dev)
Checking /sys/sun4/conf/files...               (add sundev/md.c, sundev/mdsubr.c to files, if necessary)
sundev/md.c appended to /sys/sun/../sun4/conf/files
sundev/md_subr.c appended to /sys/sun/../sun4/conf/files
Checking /usr/include/sun/dkio.h...            (4.0.3 only: checking if necessary defines in dkio.h)
Checking /sys/sun/dkio.h...                    (4.0.3 only: checking if necessary defines in dkio.h)
Checking /usr/include/sys/buf.h...             (4.0.3 only: checking if necessary defines in buf.h)
Checking /sys/sun/../sys/buf.h...              (4.0.3 only: checking if necessary defines in buf.h)
Moving man pages...                            (moving man pages to /usr/man, if it exists)
Done.
#
```

Files Affected by MetaDisk Driver Installation

The following files are modified by `metascript`, in preparation for the kernel build. Note that only two files are affected on SunOS 4.1 system.

SunOS 4.0.3 System

```
/sys/sun4/OBJ/ufs_vfsops.o
/sys/sun4/OBJ/vfs_bio.o
/sys/sun4/conf/files
/sys/sun/conf.c
/sys/sun/dkio.h
/sys/sys/buf.h
```

SunOS 4.1 System

```
/sys/sun4/conf/files
/sys/sun/conf.c
```

Making a GENERIC MetaDisk Kernel

This procedure will create a GENERIC kernel (GENERIC.md), with MetaDisk Driver support. See the following subsection if you want to customize the kernel further, or if you want to add MetaDisk Driver support to an existing kernel.

1. Become root.
2. Copy the GENERIC configuration file to another name (GENERIC.md in the example below):

```
# cd /sys/sun4/conf
# cp GENERIC GENERIC.md
```

3. Edit the kernel configuration file (GENERIC.md, in this case) by adding the line for MetaDisk Driver support in the area of the other pseudo-devices:

```
 . . .
#
# Include support for all possible pseudo-devices.
#
# The first few are mostly concerned with networking.
# You should probably always leave these in.
#
pseudo-device   pty             # pseudo-tty's, also needed for SunView
pseudo-device   ether           # basic Ethernet support
pseudo-device   loop            # loopback network - mandatory

#
# Include support for Metadisk Driver.
#
pseudo-device md24  #metadisk driver
 . . .
```

4. After saving the configuration file, `config` and `make` the new kernel. See the *System and Network Administration Manual* for details about kernel configuration. Substitute another name if you want to call the kernel something other than "GENERIC.md."

```
# cd /sys/sun4/conf
# config GENERIC.md                 (configure the new kernel)
# cd ../GENERIC.md
# make                              (make the new kernel)
# mv /vmunix /vmunix.old            (save the old kernel)
# cp vmunix /vmunix                 (position new kernel for booting)
```

5. Halt the system and reboot the new kernel.

Kernel Must Be /vmunix

Note that IOadmin requires the running kernel to be `/vmunix`. If you run the server kernel under a different name (e.g., `/vmunix.other`), the IOadmin tools will not be able to monitor the server.

6. Perform post-installation procedures, described in Section 2.6 below.

Making a Customized Kernel

If you want to use your own customized kernel, or if you want to modify the GENERIC kernel further than in the above example, you can add MetaDisk Driver support by adding one line to a kernel configuration file (near other pseudo-devices, normally):

```
pseudo-device md24  #metadisk driver
```

You can make other changes to the kernel configuration file at the same time; the procedure for eliminating non-existent disk devices is shown below, for example.

Eliminating Non-Existent Devices

With SunOS 4.0.3 you can view statistics for only four disks in `dmt`; with SunOS 4.1, the limit is 10. Further, you may not be able to view statistics for even four disks under SunOS 4.0.3, unless you reconfigure your kernel, eliminating non-existent disk drives. If you have SCSI drives `sd0`, `sd2`, and `sd4`, for instance, you will only be able to view statistics for the first two drives, since the kernel will have reserved space for `sd1` and `sd3`.

To eliminate the non-existent drives, comment them out of the kernel configuration file and build a new kernel.

Building and Booting the New Kernel

Then `config` and `make` the kernel as usual, `mv` the new kernel to `/vmunix`, and reboot. Note that the MetaDisk Driver must be entered exactly as shown (with a comment following a #, if you wish):

```
pseudo-device md24
```

This procedure will create a customized kernel (CUSTOM.md), with MetaDisk Driver support.

1. Become root.
2. Copy the configuration file you wish to use to another name:

```
# cd /sys/sun4/conf
# cp CUSTOM CUSTOM.md
```

3. Modify the kernel configuration file (CUSTOM.md, in this case) by adding the line for MetaDisk Driver support:

```
 . . .

#
# Include support for all possible pseudo-devices.
#
# The first few are mostly concerned with networking.
# You should probably always leave these in.
#
pseudo-device   pty             # pseudo-tty's, also needed for SunView
pseudo-device   ether           # basic Ethernet support
pseudo-device   loop            # loopback network - mandatory

#
# Include support for Metadisk Driver.
#
pseudo-device md24  #metadisk driver
 . . .
```

4. After saving the kernel configuration file, `config` and `make` the new kernel. See the *System and Network Administration Manual* for details about kernel configuration. Substitute your own name for the kernel if you do not wish to use "CUSTOM.md."

```
# cd /sys/sun4/conf
# config CUSTOM.md                  (configure the new kernel)
# cd ../CUSTOM.md
# make                              (make the new kernel)
# mv /vmunix /vmunix.old            (save the old kernel)
# cp vmunix /vmunix                 (position new kernel for booting)
```

5. Halt the system and reboot the new kernel.

Kernel Must Be /vmunix

Note that IOadmin requires the running kernel to be `/vmunix`. If you run the server kernel under a different name (e.g., `/vmunix.other`), the IOadmin tools will not be able to monitor the server.

6. Perform post-installation procedures, described in Section 2.6 below.

2.6. MetaDisk Post-Installation Procedures for Servers

Once you have made the new kernel and booted it on your server, you are ready to use the MetaDisk Driver, as described in Chapter 3 and Chapter 4. To be sure the MetaDisk Driver is always usable you will probably want to start the MetaDisk Driver daemon automatically during the reboot process. You may also want to have defined metapartitions activated, mirrored metapartitions synced during the reboot process, and metapartitions mounted automatically. This section advises you about moving MetaDisk Driver man pages and explains how to do the following automatically:

1. Include MetaDisk Driver command location in root's path variable.

2. Start the MetaDisk Driver daemon during system reboot.
3. Activate metapartitions during system reboot.
4. Resync mirrored metapartitions during system reboot.
5. Mount metapartitions and metamirrors during system reboot.

Include MetaDisk Driver Commands in root's Path

Since most MetaDisk Driver commands require that you be root, you might want to edit root's `.cshrc` file to include `/usr/etc` in the path variable, if it is not already included.

Moving MetaDisk Driver man Pages

MetaDisk Driver man pages are installed in `/sys/sun` when `extract_unbundled` is run, and moved to `/usr/man` when `metascript` is run, unless `/usr/man` does not exist. Move the man pages to a different location, if you wish.

Starting the Metadisk Driver Daemon During Reboot

To cause the MetaDisk Driver daemon to be automatically started during a reboot, include the daemon (`metad`) in `/etc/rc`, before `quotacheck`:

```
 . . .

mount -at 4.2                                          >/dev/console 2>&1

if [ -f /usr/etc/metad ]; then
      /usr/etc/metad                                          >/dev/console 2>&1
fi
                              (echo -n 'checking quotas: ') >/dev/console
quotacheck -a -p                                        >/dev/console 2>&1
                              (echo 'done.')            >/dev/console

quotaon -a
 . . .
```

Activating Metapartitions During Reboot

To set up (activate) MetaDisk metapartitions during a system reboot, run `metainit -r` in `/etc/rc.boot` before `fsck` in `/etc/rc.boot`. When the system is rebooted, the special file `/etc/.<meta>`, which reflects the state of all metapartitions before the system was halted, is read; all metapartitions that were active before the reboot are activated.

```
 . . .
mount -n -r /usr                                          >/dev/console 2>&1

if [ -f /usr/etc/metainit ]; then
      (echo "setting up MetaDisk partition(s)")          >/dev/console
      /usr/etc/metainit -r                               >/dev/console 2>&1
fi

if [ $1x = singleuserx ]; then
        (echo "checking / and /usr filesystems")          >/dev/console
        fsck -p -w / /usr                                 >/dev/console 2>&1
        error=$?
        what="Fsck"
else
 . . .
```

Resyncing Mirrored Metapartitions During Reboot

To cause automatic resyncing of mirrored metapartitions during a system reboot, start `metasync -r` in `/etc/rc.local` before file systems are mounted. See Chapter 3 for detailed definitions and descriptions of *syncing* and *resyncing*.

Normally you will want mirrored metapartitions to be resynced (brought into congruence by a kernel copy) after rebooting; to make this process automatic, start `metasync -r` in `/etc/rc.local`. All currently-active metamirrors, as reflected in the special file, `/etc/.<meta>` , will be resynced, unless one or more of the following is true:

- you used one of the meta commands (`metahalt`, `metafasthalt`, `metareboot`, or `metafastboot`) to halt and/or boot the system
- the pass number for the metamirror in `/etc/metatab` is 0
- one of the mirrored metapartitions has been offlined previous to the reboot.

  Unless a sync from the second metapartition to the first was being performed when the system was halted, the first metapartition will be assumed to be the good metapartition; the second metapartition will be synced from the first. Resyncing will be done in order according to pass number — lower numbers first.

  Note that since `metasync -r`, unlike `metasync`, does *not* reset an offlined metapartition, you will have to manually run `metasync` after the system is up, if you want to re-establish mirroring to an offlined metapartition. See Chapter 3 and Chapter 4 for detailed descriptions and examples.

```
 . . .
#
# hostname now set in rc.boot
#
(echo "Resyncing MetaDisk")                                    >/dev/console
/usr/etc/metasync -r                                           >/dev/console 2>&1

(echo -n 'starting rpc and net services:')      >/dev/console
if [ -f /usr/etc/portmap ]; then
                    portmap;                  (echo -n ' portmap')    >/dev/console
fi
 . . .
```

Mounting Metapartitions During Reboot

To have metapartitions and metamirrors mounted automatically during a system reboot, include metapartitions and metamirrors that you want mounted in `/etc/fstab`. Any metapartition referenced in `/etc/fstab` will be mounted, just as are physical partitions. See Chapter 4 for more information.

3

MetaDisk Driver Basics

Read this chapter to learn the basics about the MetaDisk Driver; see the last section of Chapter 2 for instructions about preparing to use the MetaDisk Driver; read Chapter 4 for more complex descriptions and examples of MetaDisk Driver use. The MetaDisk Driver, which allows you to create *metapartitions* that can be used like physical partitions, is transparent to applications software and to disk and controller hardware. The MetaDisk Driver gives you two basic capabilities:

- *Concatenation* of two to four physical partitions into one metapartition, up to two gigabytes in size.
- *Mirroring* of disk writes, providing for instant recovery from single-disk failure, and allowing online backups of file systems.

The MetaDisk Driver sits between the file system interface and the device driver interface and interprets information from both sides. After passing through the MetaDisk Driver, information is received in the expected form by both the file system and by the device drivers. The MetaDisk Driver fits into the normal `bdevsw` (block device) and `cdevsw` (character device) interface through which the kernel talks to other drivers.

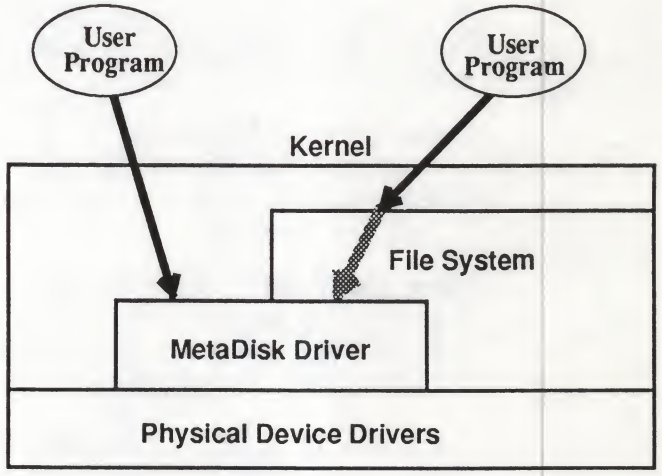


Figure 3-1 *MetaDisk Driver Position in Kernel*

Without the MetaDisk Driver, a file system is limited to one physical disk partition, and a partition can be only as large as a disk. With MetaDisk you can spread a file system over several partitions, even when the partitions are on different disk drives and different controllers. With mirroring your critical data remains online in spite of a disk failure.

3.1. When to Use MetaDisk

Use the MetaDisk for the following reasons:

- When you want to concatenate two to four physical partitions together to create a larger metapartition — up to two gigabytes. Note that concatenation tends to decrease reliability; see "Performance and Reliability Ramifications of Using MetaDisk" below.
- When you want dynamic backup of critical data. Mirroring causes reads and writes on both metapartitions in a mirrored pair. If one metapartition fails, reads and writes continue to be made on the other. You can also do online backups using mirrored data, temporarily giving up mirroring.

3.2. Overview

To use the MetaDisk Driver, you will do the following, after installing MetaDisk Driver files and configuring a MetaDisk kernel, as described in Chapter 2. Each of the steps listed below is explained in detail in later sections in this chapter.

1. Plan metapartitions — how many you need (up to 24), how big they should be (up to two gigabytes), how they should be distributed for maximum efficiency, whether mirroring is needed, etc.
2. Define metapartitions in `/etc/metatab`, using concatenation and mirroring, as appropriate.
3. Run `metainit` to activate metapartitions. Run `metastat` to see that metapartitions were set up correctly.
4. Use metapartitions just as you would physical partitions: write to the raw device, create file systems, mount file systems, include file systems in `/etc/fstab`, copy and write to file systems, etc.

Definition of Terms

See "MetaDisk Commands" below for brief descriptions of each of the MetaDisk commands. See the man page for `metatab(5)` and `metautil(8)` in Appendix A for more information about the `/etc/metatab` file and the MetaDisk commands.

metapartition: a virtual disk partition, comprised of one or more physical disk partitions, defined in `/etc/metatab`; a metapartition may be comprised of physical partitions from different disks and controllers. A metapartition might also be called a "logical volume."

metamirror: a special kind of metapartition, normally comprised of a pair of standard metapartitions, used for data mirroring; metamirrors are defined in `/etc/metatab`. A metamirror can be defined and activated with only one metapartition, in anticipation of adding a mirrored metapartition later.

/etc/metatab: the file where metapartitions are defined; similar to `/etc/fstab`; see `metainit` command below.

concatenation: the act of combining two or more physical partitions into a single metapartition.

mirroring: the act of duplicating all writes to one metapartition to a second metapartition, thus providing dynamic backup of data.

syncing/resyncing: the act of bringing mirrored metapartitions into congruence, using a "kernel copy"; see `metasync` command below.

offlined metapartition: a metamirror component which is not currently being read from or written to; a metapartition can be offlined by read or write errors, or by the `metaoffline` command. An offlined metapartition is still associated with the metamirror and cannot be used in another metapartition until it has been cleared with `metaclear`.

attach (a metapartition): to attach a metapartition to a metamirror with the `metattach` command. The metapartition will be offlined after it is attached. Normally, the next step after attaching a metapartition is to sync it to the other metapartition.

detach (a metapartition): to detach a metapartition from a metamirror with the `metadetach` command. The metapartition will no longer have any connection to the metamirror. A metapartition is normally detached from a metamirror for the purpose of doing online backups, as described in Chapter 4.

dynamic backup: the result of mirroring. Because writes are made to two metapartitions simultaneously, data is, in effect, automatically backed up when mirroring is active.

online backup: the act of backing up data without halting the system or restricting user access; this is accomplished by detaching one of a pair of mirrored metapartitions from its metamirror temporarily. The backup can be made to disk or tape.

hot spare: a disk partition which is up and running, but not being used, ready to be substituted for a failed partition which is a component of a mirrored metapartition; see `metareplace` below.

MetaDisk Commands

All of the MetaDisk commands are listed and described briefly below. See later sections in this chapter and the following chapter and the man pages for detailed descriptions for their use. MetaDisk Driver commands are installed in `/usr/etc`; include the full path name if `/usr/etc` is not in root's path variable.

Listed below is the complete MetaDisk command set, in summary; a key follows the list:

```
metad
metainit -a | special
metainit -r
metastat [-p special]
metaclear -a | special
metaoffline special
metareset special
metareplace special old-partition new-partition
metasync [buf_size] metamirror metapartition-to-be-synced-from
metasync -r
metattach metamirror metapartition-to-be-attached
metadetach metamirror metapartition-to-be-detached
metahalt
metafasthalt
metaboot
metafastboot
```

KEY:
special is the name of a metapartition
-a signifies all metapartitions
-r signifies reboot, and is not used interactively, but in a file such as `rc.boot` or `rc.local`
metamirror is the name of a metamirror
arguments separated by "|" are required — one or the other
arguments within "[]" are optional — include them or not

metad: the MetaDisk daemon; must be active before MetaDisk is used. The daemon is automatically started the first time `metainit` is run. It should be started in `/etc/rc` during system reboot.

metainit: the command used to set up or activate metapartitions and metamirrors defined in `/etc/metatab`; metapartitions cannot be used until they are activated with `metainit`.

metainit -r: inserted in `rc.boot`; during a system reboot, this command reads the special file, `/etc/.<meta>`, and activates metapartitions that were active before the reboot.

metastat: the command used to query the status of active metapartitions. With the `-p` flag, `metastat` displays metapartitions in `metatab` format; if the `metatab` entry for an active metapartition is deleted, use `metastat -p` to reconstruct the entry.

metaclear: the command used to de-activate metapartitions and metamirrors; `metaclear` is the opposite of `metainit`. Metapartitions cannot be cleared if they are "busy," and they cannot be used once they are cleared. You must clear a metamirror before you can clear its component metapartitions.

metaoffline: the command used to take a mirrored metapartition offline; it causes the metamirror to stop reading from or writing to the metapartition. The metapartition is still attached to the metamirror and cannot be used for another purpose unless it is detached with `metadetach`. You cannot offline the last metapartition of a metamirror. This command should not be used casually, since it may cause two mirrored metapartitions to become out of sync; see the warning about `metareset`, which is the opposite of `metaoffline`.

metareset: the command used to bring a mirrored metapartition online after it has been offlined due to read or write errors or by `metaoffline`. This command should *not* be used in normal circumstances, and never by a novice user; misuse of this command can cause system crashes. See "Cautions and Admonitions" in Chapter 4.

metasync: the command used to re-sync mirrored metapartitions after a disk or system failure, if the two were not resynced automatically by `metasync -r`. Also used to sync one metapartition of a metamirror to the other, after the second metapartition is added with `metattach`.

metasync -r: inserted in `rc.local`; during a system reboot, this command syncs metapartitions that were being mirrored before the system was rebooted, unless the no-resync flag (0) was set for the metamirror. Resyncing is done from the first metapartition to the second metapartition, unless a `metasync` from the second to the first was taking place when the system was halted.

metattach: the command used to attach a second metapartition to a metamirror. Can only be used if only one metapartition is attached to the metamirror. The newly-attached metapartition will be offlined; you will normally want to run `metasync` to sync this metapartition to the first metapartition of the metamirror.

metadetach: the command used to detach a metapartition from a metamirror; can only be used if two metapartitions are active. This command breaks the connection between the metapartition and its metamirror, but does not clear the metapartition. Use this command with caution, since you will probably have to resync the two metapartitions if you re-attach the metapartition later; this is the procedure used to perform online backups, as described in Chapter 4.

metareplace: the command used to replace a physical partition within one of the metapartitions of a metamirror. Use this command when a physical partition fails and you have a spare physical partition with which to replace it. The replacement is done online and is transparent to users. Note that `metareplace` modifies `/etc/metatab`, replacing the old physical partition with the new in the definition of the metapartition. You will normally want to run `metasync` after replacing the bad partition.

meta halt and boot commands: the following commands (`metahalt`, `metafasthalt`, `metareboot`, and `metafastboot`) are the same as

`halt`, `fasthalt`, `reboot`, and `fastboot`, with one difference: a flag is set, indicating that `metasync` is not needed. Use these commands in place of the standard commands if you want to reboot without resyncing mirrored metapartitions. Otherwise, if you use the standard boot or halt commands and you have `metasync -r` in `/etc/rc.local`, mirrored metapartitions will be assumed to be out of sync and will be resynced automatically — unless one of the metapartitions has been offlined previously.

metahalt: a special version of the UNIX `halt` command; used to halt the system without necessitating resyncing mirrored metapartitions during a subsequent reboot.

metafasthalt: a special version of the UNIX `fasthalt` command; used to halt the system without necessitating resyncing mirrored metapartitions during a subsequent reboot.

metareboot: a special version of the UNIX `reboot` command; used to reboot the system without resyncing mirrored metapartitions.

metafastboot: a special version of the UNIX `fastboot` command; used to reboot the system without resyncing mirrored metapartitions.

3.3. Planning Metapartitions

Before using the MetaDisk Driver to define and activate metapartitions, figure out what metapartitions you need to create — using concatenation and mirroring — and determine what physical partitions are available.

Use the IOadmin tools, `fsm` and `dmt`, the system commands, `dkinfo` and `format`, and other system commands to determine what physical partitions are defined and in use. You will find `fsm` especially helpful in planning metapartitions, since it gives you a good picture of your physical partitions and how they are being used. It is also handy in identifying potential problems: `fsm` will indicate if you have syntax problems in `/etc/metatab`, even if you haven't activated the metapartitions. You can view each disk and its partitions, for instance, with `fsm`:

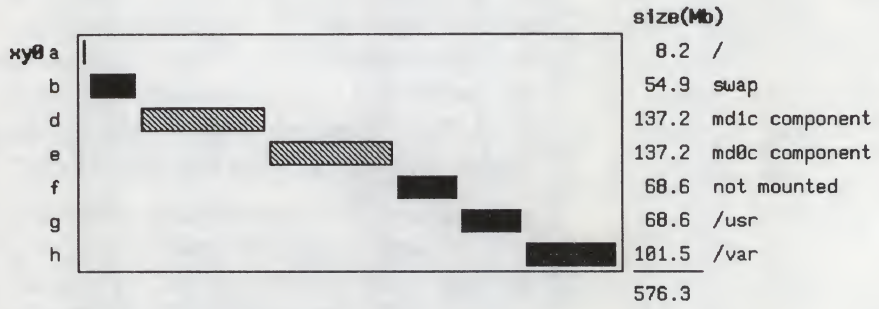


Figure 3-2 *FSM: Disk Layout*

Within the limitations of the MetaDisk, you can set up your metapartitions (up to a total of 24) almost any way you wish, but there are two rules of thumb which should guide you:

- Mirrored metapartitions should generally be on different physical disks, and on different disk controllers, if possible, for maximum safety and performance. Defining mirrored metapartitions on the same disk is a waste of disk space, since one of the primary purposes of mirroring is availability, and both metapartitions would fail if the physical disk failed.
- The primary purpose of concatenation is to create a partition larger than a single disk; define concatenated metapartitions comprised of entire disks (partition c), if possible. Note that concatenating across multiple disks and controllers *increases* the risk of failure, since failure of any one disk will necessitate recreating the damaged file system with `newfs` and restoring from backup media; do not concatenate frivolously.

Performance and Reliability Ramifications of Using MetaDisk

The MetaDisk driver allows you to mix controllers and disks of different types into a single virtual device. This gives you a lot of flexibility in the organization and use of storage, but the performance impacts can be significant. While we have general "rules of thumb" regarding performance, it is not possible to provide meaningful parametric predictions for the nearly infinite possible configurations of metapartitions and physical devices.

- Concatenation's effect on performance is quite variable, particularly with raw access and mixed disk or controller types. In typical file system use with uniform disks, concatenated metapartitions will provide slightly lower performance than the disks themselves.
- Concatenation of partitions from different disks decreases reliability, because more disks are involved; a hard failure (one that crashes the disk) on any component of a concatenated metapartition renders the metapartition useless. You will have to do a restore to recover the lost data. Concatenation of partitions from the same disk has the same effect as increasing the size of a physical partition: because the metapartition is bigger, there is more of a chance that a bad block will be encountered.
- Mirroring increases availability, especially if the second metapartition is on a different disk than the first; availability is increased further if the second metapartition is on a different controller than the first.
- Mirroring decreases performance somewhat, especially if the second metapartition is on the same disk and/or controller as the first; performance is affected least if the two metapartitions are on different controllers. Note that performance on reading is typically better than on the underlying disks, and performance on writing is typically significantly slower than writing to the disks directly.

Naming Metapartitions

You can define up to 24 metapartitions and metamirrors in `/etc/metatab`; their names are of the form `/dev/md0a`, and in the range of `/dev/md0a` to `/dev/md0h`, `/dev/md1a` to `/dev/md1h`, and `/dev/md2a` to `/dev/md2h`:

```
/dev/md0a       /dev/md1a       /dev/md2a
/dev/md0b       /dev/md1b       /dev/md2b
/dev/md0c       /dev/md1c       /dev/md2c
/dev/md0d       /dev/md1d       /dev/md2d
/dev/md0e       /dev/md1e       /dev/md2e
/dev/md0f       /dev/md1f       /dev/md2f
/dev/md0g       /dev/md1g       /dev/md2g
/dev/md0h       /dev/md1h       /dev/md2h
```

Raw Device Names

Although there are also "raw" device names in `/dev` for each of the 24 metapartitions (`/dev/rmd0a`, `/dev/rmd0b`, etc.), use the block device name (`/dev/md0a`, for example) when defining metapartitions in `/etc/metatab`.

Partitions a, b, c Have No Special Meaning

Note that the `a`, `b`, and `c` partitions do not have any special meaning, as they do, by convention, in the naming of physical partitions. Note also that there is a special device, `/dev/rmdd`, which is reserved for administrative use by MetaDisk utilities.

3.4. Defining Metapartitions: the metatab File

Once you have determined what metapartitions you want, edit `/etc/metatab` to define the desired metapartitions and metamirrors. The rules for metapartition definition are simple; each line in `/etc/metatab` is of the form

special rows `1` *real1* `1` *real2*

or

metamirror `-m` *special* `[`*special*`]` `[options]`

where the fields have the following meaning:

special — the name of the metapartition; `/dev/md0a or /dev/md0b`, for example. In the definition of a metamirror, the first *special* is the first metapartition; the second *special* (optional) is the metapartition that mirrors the first.

rows — the number of physical partitions that comprise the concatenated metapartition; a concatenation can be made up of one to four physical partitions totalling up to two gigabytes of usable capactity.

1 — precedes each physical partition. Serves as a separator, only; reserved for future functionality.

real1, real2, real3, real4 — represents the physical partition that comprise a metapartition; `/dev/xy0c`, `/dev/xy1c`, `/dev/sd0e`, `/dev/sd1e`, `/dev/id000c`, `/dev/id001c`, for example.

metamirror — a special type of metapartition, used to define a pair of mirrored metapartitions. A metamirror can be defined with only one metapartition, in anticipation of adding a mirrored metapartition later.

options — flags that are used to define a mirrored metapartition. The following options can follow the definition of a metamirror:

-e number says to take a metapartition offline if the specified read-error threshold is exceeded; note that a metapartition will be taken offline when one *write* error occurs. The default read-error threshold is zero, meaning that one read error will cause an error condition.

-g specifies read data on the first half of a metamirror from the first metapartition, and read data on the second half of the metamirror from the second metapartition. This option, which should improve read performance, is meant to be used when the first metapartition and the second metapartition are comprised of an entire disk (`/dev/id000c` or `/dev/xy0c`, for example). If each of the mirrored metapartitions contain 10000 blocks, for instance, reads of blocks 1 to 5000 will be made from the first metapartition and reads of blocks 5001 to 10000 will be made from the second metapartition.

-r specifies read only from the first metapartition; by default, reads are done from the first and second metapartition alternately.

-s specifies serial writes; write to the second metapartition after the first write has completed. By default, writes are done in parallel.

number designates the pass number (from 0 to 9) to be used for mirror resyncing during a reboot; resync will be done first for metamirrors with smaller pass numbers. Note that you can give the same pass number to a number of metamirrors. To avoid the performance degradation of having all of the mirror resyncs going on at once, you will probably want to use more than one pass number.

A pass number of 0 means to skip mirror resync during reboot. If no pass number is provided, it defaults to 1.

Sample metatab File

In the following sample `metatab` file, each metapartition definition is followed by a description of the metapartition; such descriptions are not included in an actual `metatab`. See Chapter 4 for more `metatab` examples.

```
/dev/md0a 1 1 /dev/id000c
  (simple metapartition: one physical partition)
/dev/md0b 2 1 /dev/id001g 1 /dev/id001h
  (concatenated metapartition: two physical partitions from same disk)
/dev/md0c 2 1 /dev/id002c 1 /dev/id003c
  (concatenated metapartition: two full disks)
/dev/md0d -m /dev/md0e /dev/md0f -e 4 1
  (metamirror: /dev/md0f mirrors /dev/md0e; metapartition offlined after four read errors; resync pass number = 1)
/dev/md0e 1 1 /dev/id004c
  (mirrored metapartition of /dev/md0d: one physical partition)
/dev/md0f 1 1 /dev/id005c
  (mirrored metapartition of /dev/md0d: one physical partition)
/dev/md0g -m /dev/md0h /dev/md1a -g 2
  (metamirror: /dev/md1a mirrors /dev/md0h; reads done from /dev/md0h or /dev/md1a depending on data location on
   metapartition; resync pass number = 2)
/dev/md0h 1 1 /dev/id006c
  (mirrored metapartition of /dev/md0g: one physical partition)
/dev/md1a 1 1 /dev/id010c
  (mirrored metapartition of /dev/md0g: one physical partition — on second controller)
```

3.5. Activating Metapartitions with metainit

Once you have defined metapartitions in `/etc/metatab`, you are ready to *set up* or *activate* the metapartitions, thus making them usable. The command `metainit -a` reads the `/etc/metatab` file and activates all legal metapartitions, unless they are already active. Note that `metainit` sets up a data structure in the kernel; it doesn't do anything to the physical partition(s) that comprise a metapartition, nor does it affect the data on the physical partition(s). See "Activating Metapartitions During Reboot" in Chapter 2 for instructions for activating metapartitions automatically.

Using fsm to Prevent Problems

Before activating any metapartitions, use `fsm` to make absolutely certain that you haven't done any of the following:

- used partition `a`, `b`, or `c` on the *system disk* as a component of a metapartition
- used a `c` disk partition as a component of a metapartition when a file system has been created somewhere on that disk
- used any partition that is already in use for a database or other "raw" activity as a component in a metapartition. Note that many database systems and other applications use raw partitions that may appear to be unused, but that contain valuable data; double-check before you activate any metapartition. Although activating the metapartition will not, in itself, cause any damage, any use of the metapartition (by raw or file system access) can result in data loss.

How You Can Use Metapartitions

Once a metapartition is activated, you can use it just as you would any physical partition (except for running programs such as `format` on it). If you create a file system on a metapartition, for instance, you can:

- mount the metapartition on a directory
- unmount a mounted metapartition (if it is not busy)
- copy files to the metapartition
- read and write files on the metapartition
- perform dumps from the metapartition and restore to the metapartition.

Note that if you have set up mirrored metapartitions, any writes or deletes you perform on the first metapartition will also happen on the second metapartition. Shown below is a graphic illustration of a metapartition (`/dev/md0b`) which is a concatenation of physical partitions `/dev/id001c` and `/dev/id002c`; as implied by the picture, a metapartition is a "virtual disk" and can be used like a physical disk.

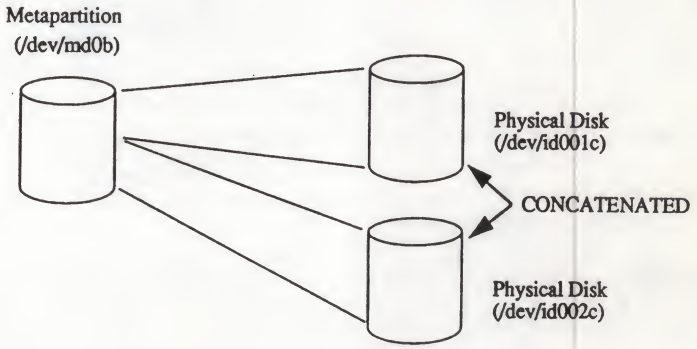


Figure 3-3 *Concatenated Metapartition*

Activating All Metapartitions at One Time

To activate all metapartitions in `/etc/metatab`, as root type `metainit` with the `-a` flag. Suppose you have the following entries in `/etc/metatab`, for example:

```
/dev/md0e 2 1 /dev/sd0a 1 /dev/xy1a
/dev/md0f 2 1 /dev/xy1f 1 /dev/sd0f
/dev/md0g 1 1 /dev/xy1b
```

When you run `metainit -a`, all of the defined entries will be activated:

```
# metainit -a
special /dev/md0e is set up
/dev/md0e:
        2 rows, total blocks 163447
        row 0:  width 1 blocks 16048    devices:        < /dev/sd0a >
        row 1:  width 1 blocks 147399   devices:        < /dev/xy1a >

special /dev/md0f is set up
/dev/md0f:
        2 rows, total blocks 178680
        row 0:  width 1 blocks 147400   devices:        < /dev/xy1f >
        row 1:  width 1 blocks 31280    devices:        < /dev/sd0f >

special /dev/md0g is set up
/dev/md0g:
        1 rows, total blocks 147400
        row 0:  width 1 blocks 147400   devices:        < /dev/xy1b >
```

As long as there are no illegal entries in `/etc/metatab`, all defined metapartitions and metamirrors will be activated; a message will be displayed for each metapartition set up. If a metapartition is already activated, the following pair of messages will be displayed for it.

```
MD_IOCSET: Permission denied
special /dev/md0b is already set up
```

Activating One Metapartition at a Time

You can activate a single metapartition defined in `/etc/metatab` by providing the name of the metapartition after `metainit`:

```
# metainit /dev/md0a
```

The same types of informational and error messages will be displayed as for `metainit -a`. Note that you cannot activate a metamirror unless its component metapartitions are activated first.

Error Conditions

Note the following types of errors which will cause `metainit` to fail. The list is not comprehensive, but includes the most common errors.

Non-Existent Metapartition

A non-existent metapartition (`/dev/md4a`) used in a metapartition definition will cause `metainit` to fail:

```
/dev/md4a 1 1 /dev/id000a
```

```
# metainit /dev/md4a
/dev/md4a: No such file or directory
```

Non-Existent Physical Partition

A non-existent physical partition (`/dev/id222a`) used in a metapartition definition will cause `metainit` to fail:

```
/dev/md0a 1 1 /dev/id222a
```

```
# metainit /dev/md0a
/dev/rid222g: No such file or directory
metainit: bad device /dev/id222g
metainit: /dev/md0a not set up
```

Physical Partition Used Twice

If the same physical partition (`/dev/id000c`) is used in definitions of two or more metapartitions, `metainit` will fail:

```
/dev/md0a 1 1 /dev/id000c
/dev/md0b 1 1 /dev/id000c
```

```
# metainit -a
metainit: device 0x1600 appears in both /dev/md0a and /dev/md0b
Can't include real device /dev/id000c in two meta-partitions
metainit: /dev/md0b not set up
```

Metapartition Name Used Twice

If the same metapartition name (`/dev/md0a`) is used to define two or more metapartitions, `metainit` will fail:

```
/dev/md0a 1 1 /dev/id000a
/dev/md0a 1 1 /dev/id000b
```

```
# metainit -a
Ioctl MD_IOCSET failed: Permission denied
special /dev/md0a is already set up
```

Not Enough Rows Defined

If fewer physical partitions are defined (1) than indicated by the rows argument (2), `metainit` will fail:

```
/dev/md0a 2 1 /dev/id000a
```

```
# metainit /dev/md0a
metainit: 2 remaining arguments not enough for 2 remaining rows
metainit: errors in line: '/dev/md0a 2 1 /dev/id000a': /dev/md0a not set up
```

Activating Metamirror Before Metapartitions

If you try to activate a metamirror when no component metapartitions are activated, `metainit` will fail. Assume the following metamirror and metapartitions have been defined in `/etc/metatab`, but have not yet been activated with `metainit`;

```
/dev/md0a -m /dev/md0b /dev/md0c
/dev/md0b 1 1 /dev/id000a
/dev/md0c 1 1 /dev/id001a
```

If you run `metainit -a`, the component metapartitions will be activated before the metamirror, so there is no problem. If you try to activate the metamirror by itself with the command, `metainit /dev/md0a`, however, the action will fail, with an indication that the mirror unit is not set up. You will have to activate the metamirror components first.

```
# metainit /dev/md0a
md: mirror unit 4 not set up
Ioctl MD_IOCSET failed: No such device or address
Can't set up special /dev/md0a
```

3.6. Checking the Status of Metapartitions

To see descriptions of active metapartitions, use the `metastat` command. You should also use `fsm` as a cross check for `metastat`; make sure you have set up what you thought you did *before* you try to use it.

You do not need to be root to run `metastat`. With no arguments, `metastat` displays the status of all active metapartitions. Suppose you have activated all the partitions in the following `metatab`:

```
/dev/md0a 2 1 /dev/xy1g 1 /dev/xy1h
/dev/md0b -m /dev/md0c /dev/md0d -e 10 2
/dev/md0c 3 1 /dev/sd0d 1 /dev/sd0b 1 /dev/sd0h
/dev/md0d 1 1 /dev/xy0g
```

Run `metastat` to see the status of each of the metapartitions and metamirrors:

```
% metastat
*****
/dev/md0a:
    2 rows, total blocks 394800
    row 0:  width 1 blocks 147400   devices:    < /dev/xy1g >
    row 1:  width 1 blocks 147400   devices:    < /dev/xy1h >
*****
/dev/md0b: metamirror
    first mirror partition is /dev/md0c
    second mirror partition is /dev/md0d
    error threshold = 10   pass = 2
    alternate reads between the first and second mirror partitions
    dispatch writes to the first and second mirror partition in parallel
   /dev/md0c: (first)
    3 rows, total blocks 108528
    row 0:  width 1 blocks 25024    devices:    < /dev/sd0d >
    row 1:  width 1 blocks 52224    devices:    < /dev/sd0b >
    row 2:  width 1 blocks 31280    devices:    < /dev/sd0h >

   /dev/md0d: (second)
    1 rows, total blocks 147400
    row 0:  width 1 blocks 147400   devices:    < /dev/xy0g >

*****
```

In the above example, one concatenated metapartition (`/dev/md0a`) and one metamirror (`/dev/md0b`) are displayed. Simple metapartitions (`/dev/md0a`) display the number of rows, how many blocks in the metapartition, and which physical partition comprises each row.

A metamirror (`/dev/md0b`) lists the metapartitions which are components of the metamirror, with information about error thresholds, pass numbers, etc., followed by descriptions of the physical partitions that comprise the mirrored metapartitions. In the above example, the metapartition components (`/dev/md0c` and `/dev/md0d`) of the metamirror, `/dev/md0b`, are not the same size; the metamirror will be the size of the smaller of the two metapartitions (`/dev/md0c`).

To query the status of a single metapartition (metamirror, mirrored metapartition, or concatenated metapartition), include the name of the metapartition after `metastat`:

```
% metastat /dev/md0c
/dev/md0c: (mirror)
        3 rows, total blocks 108528
        row 0:  width 1 blocks 25024    devices:        < /dev/sd0d >
        row 1:  width 1 blocks 52224    devices:        < /dev/sd0b >
        row 2:  width 1 blocks 31280    devices:        < /dev/sd0h >
```

3.7. Clearing Metapartitions

Use the command `metaclear` to clear (deactivate) active metapartitions. An inactive metapartition is not usable; it cannot be written to or read from. With the `-a` flag `metaclear` clears all metapartitions and metamirrors.

```
# metaclear -a
```

Actually, `metaclear -a` clears all metapartitions that it can; if a metapartition is mounted, or if it is being used in some way, it will not be cleared; a message explaining why it could not be cleared will be displayed. If metapartition `/dev/md0b` is mounted or otherwise busy, for instance, the following message will be displayed if you try to clear it:

```
Ioctl MD_IOCRESET failed: Device busy
metaclear: command failed
```

This message means one of two things:

- the file system on the metapartition is mounted
- the metapartition (either raw or block) is open

Clearing a Single Metapartition

To clear a single metapartition, provide the name of the metapartition:

```
# metaclear /dev/md0b
```

The same rules apply as for clearing all metapartitions with `metaclear -a`; if the metapartition is busy, it will not be cleared.

4

MetaDisk Driver Tutorial

Read this chapter to learn more about MetaDisk Driver, and to see examples of various types of metapartitions. Be sure you have read and understood all of this chapter, especially the "Cautions and Admonitions" below, before you use the MetaDisk Driver. For more information, see Chapter 3 and the man pages for `metatab(5)` and `metautil(8)`.

4.1. Summary: Installing and Using Metadisk Driver

Read this section for a summary of what you have to do to install and use the MetaDisk Driver. See Chapter 3 and the rest of this chapter for detailed background and instructions.

Installing Software and Building Kernel

Do the following to prepare to use MetaDisk Driver:

1. Install software from tape (Chapter 2).
2. Build a MetaDisk Driver kernel (Chapter 2).
3. Include MetaDisk Driver daemon startup in `/etc/rc` (Chapter 2).
4. Include MetaDisk directory in root's command path (Chapter 2 — optional).
5. Reboot with MetaDisk Driver kernel in `/vmunix` (Chapter 2).

Defining and Activating Metapartitions

Do the following to define and activate metapartitions and metamirrors:

1. Define metapartitions in `/etc/metatab` (Chapter 3, Chapter 4).
2. Activate metapartitions with `metainit` (Chapter 3, Chapter 4).
3. Check status of metapartitions with `metastat` (Chapter 3, Chapter 4).
4. Make metapartition setup during reboot automatic with `metainit -r` and `metasync -r` (Chapter 3, Chapter 4).

Making and Using File Sytems on Metapartitions and Metamirrors

Once you have defined and activated metapartitions and metamirrors, do the following to use them as file systems:

1. Use `newfs` to make a file system on the metapartition or metamirror (Chapter 4).

   If you have defined a metamirror, make the file system on the metamirror and *not* on the metapartitions that comprise the metamirror.

2. Mount the file system (Chapter 4).
3. Include the file system mount in `/etc/fstab` (Chapter 4 — optional).
4. Use the file system just as you use any other file system.

4.2. Cautions and Admonitions

Note the following when using metapartitions and metamirrors. Do not use the MetaDisk Driver until you have read and understood this section. See Section 4.9, "Safety Tips," also.

- **Do not concatenate frivolously.**

  Because concatenation decreases reliability and makes recovery more complicated, do not concatenate unless it is necessary. Ideally, every metapartition will be comprised of an entire physical disk (partition `c`) and concatenation will not be necessary. If you need to concatenate in order to create a larger metapartition, be aware of the increased exposure in case of a disk failure, and be aware that recovery will be more complicated.

- **Don't try to kill the MetaDisk Driver daemon.**

  The MetaDisk Driver daemon (`/usr/etc/metad` must be running if you wish to use the MetaDisk. It is started automatically the first time `metainit` is run.

- **Do not remove the special file `/etc/.<meta>` .**

  If this special file is removed and you have a metapartition defined for mounting in `/etc/fstab`, reboot will fail if `metainit -r` is in `rc.boot`. The file `/etc/.<meta>` is created when `metainit` is run. It contains the state information used by `metainit -r` and `metasync -r` to activate metapartitions and sync mirrored metapartitions.

- **Use `newfs` with care.**

  Running `newfs` on a metapartition is just like running `newfs` on a physical partition: any data that was on the metapartition will be lost once you run `newfs`.

- **Do not try to boot from a metapartition.**

  Metapartitions and metamirrors are not bootable. Do not mount a metapartition or metamirror on /.

  Note that if you run SunOS 4.0.3 `newfs` on metapartition `a` (e.g., `/dev/rmd0a, /dev/rmd1a, /dev/rmd2a`), you will see error messages related to installation of boot code; this happens because SunOS 4.0.3 `newfs` tries to install a bootblock on any partition `a`. This is a harmless condition; ignore the messages.

- **Do not mount a metapartition on `/usr`.**

  Since the MetaDisk commands are on `/usr`, you cannot mount a metapartition on it.

- **Metapartitions must be activated before being used.**

  Metapartitions are not usable until they have been activated with `metainit`; `metainit` reads `/etc/metatab` to determine what metapartitions can be activated. Activated metapartitions can generally be treated just like physical partitions.

- **Act on metamirrors or metapartitions only, once activated.**

  Once you activate a metamirror or metapartition, you should only act on the metamirror or the metapartition — not on the physical partition or partitions that comprise the metapartition. Commands such as `newfs`, `mount`, and `umount` should be performed on the metamirror only, if you have defined a metamirror; for simple, concatenated metapartitions, perform these commands on a metapartition only.

  Never mount a physical partition that is a component of a metapartition. There is no way for the MetaDisk Driver to know if you do this, so no error message will be displayed; check and double-check before mounting a physical partition that might be part of a metapartition.

  Note: if you have defined a pair of mirrored metapartitions, using *one* physical partition per metapartition, and you decide you do not want to use the mirroring any more, detach one of the mirrored metapartitions with `metadetach` and clear it with `metaclear`. After running `fsck` against it, you can mount the physical partition that comprised the metapartition; no data will be lost.

  Be sure to remove the metapartition definition from `/etc/metatab`, to prevent potential disaster. Also, be sure you do not perform this detach procedure on a concatenated metapartition; the data on the physical partitions that comprised the metapartition will not be usable. You would have to `newfs` the physical partitions and restore the data from a backup copy.

- **Do not write on the first block of a disk.**

  Be careful with "raw" writes to the first block of a metapartition if that metapartition's first block is the first block of a physical disk. The disk label is always written in the first block of a physical disk, and writing to that block will destroy the disk label, requiring that you reformat, partition, and label the disk before restoring from backup.

  If the first block of a physical disk appears anywhere else than the first block of a metapartition (if it is the second, third, or fourth partition in a concatenated metapartition, in other words), the block is skipped automatically; you do not need to worry about these cases. You do not to worry about the first block of the physical disk if the metapartition is used for a file system, either, since the first block is protected automatically, just as it is with file systems on physical disk partitions.

- **Use block or raw device names, as appropriate.**

  Use the block device name (`/dev/md0a`, etc.) for block device functions such as `mount` and `umount`; use the raw device name (`/dev/rmd0a`, etc.) for raw device functions such as `newfs` or `dd`.

- **Do not try to create multi-layered metapartitions.**

  A metapartition cannot be defined with another metapartition as a component; only physical partitions can be used as components of a metapartition.

- **Do not try to clear mounted metapartitions.**

  A mounted metapartition cannot be deactivated with `metaclear` until it has been unmounted. Note that a metapartition, just like a physical partition, cannot be unmounted if it is busy.

- **Monitor metamirrors carefully.**

  When you have defined and activated a pair of mirrored metapartitions, it is important that you check for failure of one of the metapartitions. Observe console messages, run `metastat`, check `dmt` error logs, etc. As soon as one of the metapartitions fails, mirroring is disabled (a metapartition "fails" when there is one write failure or when read failures exceed the error threshold defined in `/etc/metatab`). You should replace the failed disk partition(s) as soon as possible, and resync.

- **Try to make mirrored metapartitions the same size.**

  Disks of differing physical size or geometry can be used effectively with the MetaDisk Driver; what is important in defining metamirrors is that the mirrored metapartitions be the same size, or nearly so. If you define a metamirror with two metapartitions of different sizes, the metamirror will be the size of the smaller of the two metapartitions; the extra space in the second metapartition will be wasted.

  If you define a metamirror with one metapartition, the metamirror will be the size of that metapartition, regardless of the size of a metapartition attached later. You will not be allowed to attach a smaller metapartition.

- **Mount a metamirror — not one of its component metapartitions or physical partitions.**

  For mirrored metapartitions, only the metamirror can be mounted; you will get an error message if you try to mount a metapartition which is a component in a metamirror. However, you will *not* get an error message if you try to mount a physical partition which is a component of a metapartition; this could destroy data and crash the system.

  When a metamirror with two metapartitions defined is activated with `metainit` and mounted on a file system, any writes to the first metapartition are also made to the second metapartition.

In the picture below, the metamirror (`/dev/md0a`) should be mounted, and not the mirrored metapartitions (`/dev/md0b` and `/dev/md0c`) nor the physical partitions (`/dev/id001c`, `/dev/id002c`, `/dev/id011c`, and `/dev/id012c`). The MetaDisk Driver will not allow you to mount the metapartition components, but MetaDisk Driver cannot prevent you from mounting the physical partitions, which could be a fatal mistake.

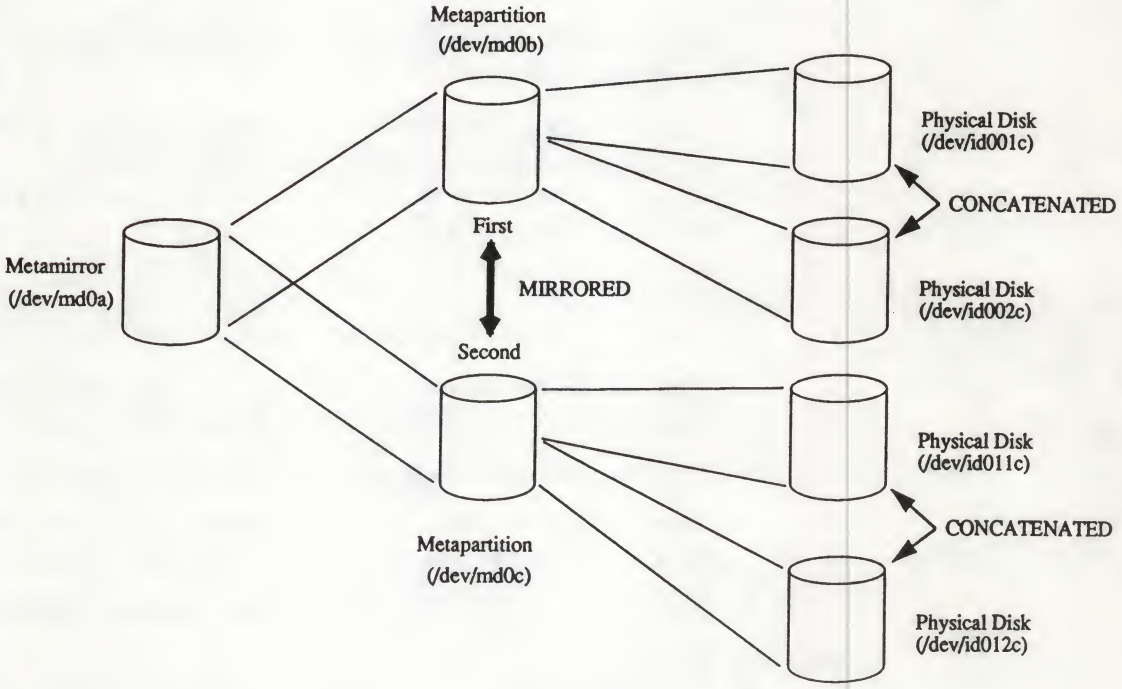


Figure 4-1 *Mirrored Metapartitions*

- **Run `fsck` after detaching a metapartition from a metamirror.**

  Because the state of a file system will be inconsistent after `metadetach`, you should always run `fsck` on the detached metapartition.

- **Resync after disk or system failure.**

  Because of the danger of a system panic caused by file systems on the two mirrored metapartitions being out of sync, you should always run `metasync` against a metamirror after one of the disks has been taken offline — either by a disk failure or by the commands `metaoffline` or `metadetach`. You should also run `metasync` against all metamirrors after a a system reboot — unless you did an orderly shutdown using one of the `meta` halt or boot commands. Typically, you should designate automatic resyncing in `/etc/rc.local`, and you should `fsck` all file systems before mounting them.

- **Do not use `metareset` carelessly.**

  The command `metareset` should rarely be used, if ever. Because of the danger of a system panic caused by file systems on mirrored metapartitions being out of sync (see above), do not use `metareset` to bring a metapartition back online for mirroring unless you are sure the two metapartitions are exactly the same. Normally you will want to run `metasync` to sync the two metapartitions after one of them has been offline.

- **Insert MetaDisk commands in startup files for automatic metapartition setup and resyncing.**

  Three MetaDisk commands must be inserted in system startup files if you want automatic rebooting and resyncing of metapartitions: `metad`, `metainit -r`, and `metasync -r`. See "MetaDisk Post-Installation Procedures" in Chapter 2 for detailed instructions.

- **When resyncing, be sure to sync from the good metapartition.**

  The `metasync` command requires you to name the metamirror and the *good* metapartition, the one from which you will be copying to the other.

- **When resyncing, start resync before using metamirror.**

  Start `metasync` before you use the metamirror for anything; `metasync` should be started before `fsck` or `mount`, for instance.

- **Do not remove entries from `/etc/metatab` if metapartition is still active.**

  To maintain consistency, you should always de-activate a metapartition that you remove from `/etc/metatab`.

- **Use `dmt` and `fsm` for disk and MetaDisk Driver information and functions.**

  Note that you can use the IOadmin tool `dmt` to look at the physical disk layout, and `fsm` to find out information about metapartitions, and to perform certain functions on metapartitions. You will probably find it more convenient to use the IOadmin tools than to use the standard UNIX commands. You can see all that you need to know about a disk partition by selecting it and clicking on "Selection Detail," for instance:

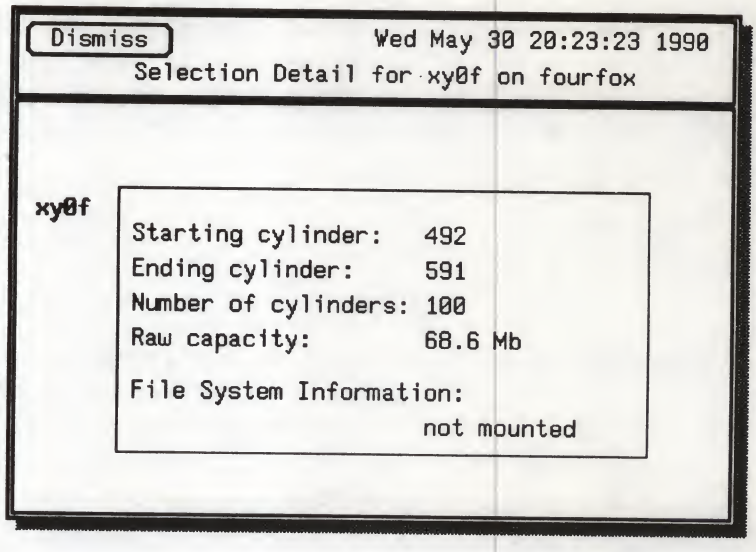


Figure 4-2 *FSM: Unmounted File System*

4.3. Defining Metapartitions

Use the following descriptions and examples in defining various types of metapartitions and metamirrors.

Defining a Simple Metapartition

Metapartitions are defined in `/etc/metatab`. There are two basic types of metapartitions: simple (concatenated) metapartitions and metamirrors. Shown below are examples of `/etc/metatab` entries for simple metapartitions; metamirrors are shown in the next subsection. Remember that once you have defined and activated a metapartition, you should always deal with the metapartition only, and not with the physical partition or partitions that comprise it; commands such as `newfs` and `mount` should be run on the metapartition only.

One Physical Partition per Metapartition

This simplest of metapartitions is identical to the underlying physical partition; this type of metapartition is seldom used by itself, but is generally used as a component of a metamirror. Note that you do not have to `dump` and `restore` to convert a physical partition into a metapartition.

```
/dev/md0a  1  1  /dev/xy0e
```

Multiple Physical Partitions (one disk)

This configuration is of limited usefulness; use it if you have two physical partitions on a disk that you wish to combine into one metapartition without repartitioning the disk.

```
/dev/md0a  2  1  /dev/xy0e  1  /dev/xy0f
```

Multiple Physical Partitions (two full disks)

This is a more useful configuration, since it combines two full disks into one metapartition, thus giving you a larger partition than you could have without MetaDisk. Note that you could combine up to four physical partitions to make up one metapartition; the practical limit for a metapartition is two gigabytes, due to UNIX limitations. This example assumes that the physical partition was defined using the UNIX convention that 'c' names the partition that defines the entire disk.

```
/dev/md0a  2  1  /dev/id000c  1  /dev/id001c
```

Creating metatab Entry with metastat

If the definition of a metapartition is inadvertently removed from `/etc/metatab`, you can reconstruct the `metatab` entry with `metastat -p`, if the metapartition is active. If the following metapartition, as shown from the `metastat` command, has been deleted from `/etc/metatab`, for instance:

```
/dev/md1a:
    2 rows, total blocks 557190
    row 0:  width 1 blocks 263220  devices:  < /dev/id001b >
    row 1:  width 1 blocks 293970  devices:  < /dev/id001h >
```

1. Run `metastat -p`, supplying the name of the metapartition:

```
# metastat -p /dev/md1a
/dev/md1a 2 1 /dev/id001b 1 /dev/id001h
```

2. Enter the result of `metastat -p` in `/etc/metatab`.

Defining a Metamirror

A metamirror, which is a special kind of metapartition, is defined in `/etc/metatab`. Shown below are examples of `/etc/metatab` entries for metamirrors. Remember that once you have defined and activated a metamirror, you should always deal with the metamirror only, and not with the physical partition or partitions that comprise it; commands such as `newfs` and `mount` should be run on the metamirror only — and *not* on the metapartitions that comprise the metamirror.

Metamirror: One Metapartition Defined

Define a metamirror with one metapartition in anticipation of mirroring the metapartition later; you will not have to halt the system or unmount the existing metapartition when you decide to activate mirroring. Use this configuration when you anticipate doing online backups of a metapartition, also. You may want to use this method by default, assuming that you may later want to add mirroring to do online backups.

```
/dev/md0a  -m    /dev/md0b
/dev/md0b  1  1  /dev/id000e
```

Note that the metamirror will be the size of the metapartition defined originally; if you add a second metapartition later, it must be at least as large as the first metapartition. The metamirror will be the size of the first metapartition, even if the second metapartition is larger.

Metamirror: Two Metapartitions, One Physical Partition per Metapartition

A very useful configuration for a metamirror is two metapartitions, each comprised of one physical partition, and with each physical partition on a separate controller.

Assume that `id000c` and `id010c` are the same size; the metamirror will be the size of the smaller metapartition if they are not the same size.

```
/dev/md0a  -m    /dev/md0b  /dev/md0c
/dev/md0b  1  1  /dev/id000c
/dev/md0c  1  1  /dev/id010c
```

Metamirror: Multiple Physical Partitions in Metapartition

Another useful configuration, despite the slight loss of reliability because of concatenation, is a metamirror with metapartitions comprised of two or more physical partitions. Remember that the metamirror will be the size of the smaller of the two metapartitions — if both are defined before the metamirror is activated.

```
/dev/md0c -m /dev/md0a /dev/md0b
/dev/md0a 3 1 /dev/id000d 1 /dev/id000e 1 /dev/id001e
/dev/md0b 1 1 /dev/id002h
```

Metamirror: -r and -s Options

In the following example, reads are only done from the first metapartition (`-r`) and writes are done serially (`-s`). Note also that the first metapartition (`/dev/md0c`) is comprised of one full Xylogics disk (`/dev/xy0c`) while the second metapartition (`/dev/md1a`) is comprised of one partition of an IPI drive (`/dev/id000d`). The metamirror will be the size of the smaller of the two physical disk partitions.

```
/dev/md0a  -m    /dev/md0c  /dev/md1a  -r  -s
/dev/md0c  1  1  /dev/xy0c
/dev/md1a  1  1  /dev/id000d
```

Metamirror: -g Option

In the following example, reads of data from the first half of a metamirror are done from the first metapartition (`/dev/md0c`), while reads of data from the second half of a metamirror are done from the second metapartition (`/dev/md1a`). Note that the `-g` option is most effective when the first metapartition and the second metapartition are each comprised of an entire disk (`/dev/id000c` or `/dev/id001c`, for example).

```
/dev/md0a  -m  /dev/md0c  /dev/md1a  -g
/dev/md0c  1  1  /dev/id000c
/dev/md1a  1  1  /dev/id001c
```

Metamirror: Read-Error Threshold (-e) and Pass Number Options

In the following example, the read-error threshold is set for four, and the pass number is set for one. A read-error threshold of four means that a faulty metapartition (either the first or the second metapartition) will be taken offline if more than four read errors are encountered. A pass number of one means that this pair of mirrored metapartitions will be among the first to be resynced in the case of a system reboot. Note that a metapartition will be taken offline when *one* write error occurs, as indicated by the device controller.

```
/dev/md0c  -m  /dev/md0c  /dev/md1a  -e 4  1
/dev/md0c  1  1  /dev/xy0c
/dev/md1a  1  1  /dev/id000d
```

Sample Metatab

Shown below is a sample `/etc/metatab`, with concatenation and mirroring combined. Note that the physical partitions that comprise the mirrored metapartitions are on different controllers than the physical partitions that comprise the first metapartitions, thus providing extra security in case of a controller failure.

Note also that the first two metamirrors have the same pass number, 1, indicating that they will be resynced simultaneously after a reboot, while the third metamirror has a pass number of 2, meaning that it will be resynced after the other metamirrors have been resynced. Doing some of the resyncs after others have finished minimizes the impact on system performance.

```
/dev/md0a 3 1 /dev/xy0c   1 /dev/xy1c 1 /dev/sd0c
/dev/md0b 2 1 /dev/id000c 1 /dev/id001c
/dev/md0c -m  /dev/md0d /dev/md0e 1
/dev/md0d 1 1 /dev/id002c
/dev/md0e 1 1 /dev/id010c
/dev/md0f -m  /dev/md0g /dev/md0h 1
/dev/md0g 2 1 /dev/id003c 1 /dev/id004c
/dev/md0h 2 1 /dev/id011c 1 /dev/id012c
/dev/md1a -m  /dev/md1b /dev/md1c 2
/dev/md1b 2 1 /dev/id005c 1 /dev/id006c
/dev/md1c 2 1 /dev/id013c 1 /dev/id014c
```

4.4. Activating, Checking, and Clearing Metapartitions

Read this section to learn how to activate metapartitions you have defined in `/etc/metatab`, how to clear active metapartitions, and how to check the status of defined metapartitions.

Activating Metapartitions

As noted in the previous chapter, metapartitions are activated by the command `metainit`. Activate all legal metapartitions in `/etc/metatab`, including metamirrors, with the `-a` option.

```
# metainit -a
```

Activating a Single Metapartition

Activate a simple metapartition, `/dev/md0a` (defined in `/etc/metatab`), as follows:

```
# metainit /dev/md0a
```

Activating a Single Metamirror

Activate a single metamirror, `/dev/md0b` (defined in `/etc/metatab` as being comprised of `/dev/md0c` and `/dev/md0d`), as follows:

```
# metainit /dev/md0c
# metainit /dev/md0d
# metainit /dev/md0b
```

Note that you cannot `metainit` the metamirror (`/dev/md0b`) until after you have activated both of the metapartitions that comprise the metamirror. Note that `metainit -a`, by making two passes of `/etc/metatab`, takes care of this automatically. It's a good idea to check the status of all metapartitions (using `metastat`) before using `metainit`.

Checking Metapartition Status

Check the status of a single metapartition as follows:

```
# metastat /dev/md0a
/dev/md0a:
  2 rows, total blocks 108528
  row 0:  width 1 blocks 25024  devices:  < /dev/id000d >
  row 1:  width 1 blocks 52224  devices:  < /dev/id000e >
```

Check the status of all metapartitions with `metastat` (no arguments).

Clearing Metapartitions

Use `metaclear` to clear metapartitions that have been activated with `metainit`; you might want to clear a metapartition so that you can redefine it, creating a larger metapartition. Note that you cannot clear a mounted metapartition, or one that is open for reads or writes; you will get a "device busy" message.

To clear all active metapartitions use `metainit -a`. To clear a single metapartition or metamirror, supply the device name of the metapartition to be cleared:

```
# metaclear /dev/md0d
```

Note that `metaclear` will *not* clear metapartitions which are components of a metamirror if you clear the metamirror; the metapartitions remain active, but they no longer have any association with the metamirror.

4.5. Modifying an Unused Metapartition or Metamirror

It is relatively easy to modify the definition of a metapartition or metamirror that has not been used — that is, metapartition or metamirror that has not been mounted or written to. You may wish to change the size of a metapartition, replace a physical partition in the definition of a metapartition, or make a metapartition a component of a metamirror, for example.

The procedure for changing a metapartition varies, depending on whether or not the metapartition has been activated. Note that activating a metapartition does not imply that it has been used; activating a metapartition does not change the physical partition that is included in the definition of the metapartition.

See "Changing the Definition of a Used Metapartition" in Section 4.7 if you wish to change the definition of a metapartition that has been written to.

If the Metapartition Has Not Been Activated

If you have defined the metapartition in `/etc/metatab`, but have not activated it with `metainit`, all you have to do is edit `/etc/metatab` file, making whatever changes you wish.

If the Metapartition Has Been Activated

If the metapartition or metamirror has been activated with `metainit`, but has not been used — that is, if it has not been mounted or written to — the process is a little more involved, but still simple:

1. Clear the metapartition.

```
# metaclear /dev/md0a
```

2. Edit the `metatab` file.

   Make whatever changes you wish to `/etc/metatab` — within the rules of the MetaDisk Driver — and save the file.

3. Activate the metapartition.

```
# metainit /dev/md0a
```

   Unless you have defined an illegal configuration, the metapartition will be set up; run `metastat` to confirm.

Adding a Second Metapartition to a Metamirror

Suppose you have defined a metamirror (`/dev/md0a`) with only one metapartition (`/dev/md0b`), and have also defined a second metapartition (`/dev/md0c`), but you have not used it. You may have activated `/dev/md0c` or not; it does not matter at this point.

```
/dev/md0a -m /dev/md0b
/dev/md0b 1 1 /dev/id000c
/dev/md0c 1 1 /dev/id010c
```

1. Define and activate the second metapartition, if necessary.

```
# metainit /dev/md0c
```

   If you haven't already done so, define the second metapartition in `/etc/metatab` (being sure the metapartition is at least as large as the first metapartition) and activate it with `metainit`.

2. Attach the second metapartition to the metamirror.

```
# metattach /dev/md0a /dev/md0c
```

   Note the syntax of `metattach`: name the metamirror, then the metapartition that is being attached.

   If you wish to sync the two metapartitions, see "Syncing Mirrored Metapartitions" in the next section for instructions.

Making a Physical Partition into a Metamirror Component

It is recommended that you plan ahead, making all your file systems run on single-partition metamirrors, thus making it very easy to attach a second metapartition later — for doing online backups or for mirroring.

If you don't do that, there may be times when you may wish to make a single, existing physical partition into a component of a metamirror — to provide mirroring or to allow you to do online backups. The process is simple, but it entails unmounting the existing partition, which may mean halting the system; the advantages of mirroring or online backups will have to be balanced against this contingency.

1. Define metapartition in `/etc/metatab`.
2. Unmount the physical partition, if necessary.
3. Activate the metapartition and the metamirror with `metainit`.
4. Edit `/etc/fstab`, if the metamirror has a file system.
5. Edit `rc.local` to add `metainit -r` and `metasync -r`, if necessary.
6. Use the metapartition

Example

Do the following to make an existing physical partition (`/dev/id000h`) into a component metapartition (`/dev/md0c`) of a metamirror (`/dev/md0a`):

1. Define the physical partition as a component of a metamirror in `/etc/metatab`. In this case, we have also defined the second metamirror component (`/dev/md0d`), but you could do this later, if you wished.

```
 . . .
/dev/md0a -m  /dev/md0c /dev/md0d
/dev/md0c 1 1 /dev/id000h
/dev/md0d 1 1 /dev/id001h
```

The metamirror, the metapartitions, and the physical partition must not be being used for any other purposes, of course.

2. Unmount the physical partition (`/dev/id000h`)

```
# umount /dev/id000h
```

You will not be able to unmount the partition if it is busy. You may have to bring the system down to single-user mode if users are using it.

3. Activate the metapartitions and the metamirror (metapartitions first).

```
# metainit /dev/md0c
# metainit /dev/md0d
# metainit /dev/md0a
```

At this point, metapartition `/dev/md0c` is congruent with physical partition `/dev/id000h`.

4. Modify `/etc/fstab`.

Remove the line that mounted the physical partition; ***this is vital***, to prevent accidentally mounting the physical partition by running `mount -a`. Insert a line to mount the metamirror, if you wish:

```
 . . .
(Delete this line)  /dev/id000h  /home  4.2 rw  1  3
/dev/md0a           /home    4.2 rw          1      3
 . . .
```

5. Set up automatic activation and resyncing.

If you haven't already done so, you will probably want to include `metainit -r` in `/etc/rc.boot` and `metasync -r` in `/etc/rc.local`, so that metapartitions will be set up and mirrored metapartitions will be synced automatically during a reboot.

6. Use the metamirror.

Use the metamirror just as you would a physical partition, mounting it, if you wish, and including the mount instructions in `/etc/fstab`. The mounted metamirror will contain the same data that the physical partition did before it was unmounted. The metamirror can now be used as the

physical partition was before this procedure.

Note that you can mount and use the metamirror before the `metasync` has completed, but be aware that mirroring to the second metapartition will not be complete until the `metasync` is completed.

Once the `metasync` has completed, you can detach the second metapartition from the metamirror and do and online backup as described in "Performing Online Backups — Mirroring Active" below.

4.6. Using Metapartitions and Metamirrors

Once you have defined and activated a metamirror or metapartition, you can do with the metamirror or metapartition whatever you might do with a physical partition (except for running programs like `format` on it).

Making and Mounting a File System

In most cases you will want to create a file system on a metapartition or metamirror you have defined and activated, and to mount the file system. Note that the directory where a file system is mounted must exist before the `mount` command is performed.

1. Make a file system on the metamirror or metapartition.

```
# newfs /dev/md0a
```

Do *not* run `newfs` on the physical disk partitions that comprise a metapartition or metamirror; do *not* run `newfs` on a metapartition which is a component of a metamirror. In general, be careful when using `newfs`; all data on the metapartition(s) will be lost. Any data on the physical partitions that comprise the metapartition(s) will be lost, also.

2. Mount the file system on a directory (`/mnt` in this case):

```
# mount /dev/md0a /mnt
```

The directory `/mnt` must exist, of course. Provide arguments to the `mount` command if you do not want the default values.

3. Include file system mount in `/etc/fstab` (optional).

```
/dev/md0a   /mnt   4.2   rw,noquota   1   4
```

Do this if you wish the file system to be mounted on the metamirror or metapartition during a system reboot, and to make unmounting and remounting it simpler.

Syncing Mirrored Metapartitions

There are times when you will need to run `metasync` to sync a metapartition to its metamirror pair — if you have created a metamirror with only one metapartition, for instance, and you later want to start mirroring. You will also need to resync if one of the metapartitions in a metamirror goes offline for any reason, and you want to resume mirroring.

Note the syntax of `metasync`: name the metamirror, then the metapartition that is being synced *from*:

```
# metasync /dev/metamirror /dev/good metapartition
```

Suppose you have defined the following metamirror (`/dev/md0a`) in `/etc/metatab`:

```
/dev/md0a -m /dev/md0b /dev/md0c
/dev/md0b 2 1 /dev/id000c 1 /dev/id001c
/dev/md0c 2 1 /dev/id004c 1 /dev/id005c
```

If the mirrored metapartitions `/dev/md0b` and `/dev/md0c` get out of sync for any reason, you can bring them back into sync with the `metasync` command. If they are out of sync because of a disk failure you may have to replace the failed disk with another before doing the resync; see "Mirroring: Replacing a Failed Disk Partition" below for details.

In the following example it is assumed that `/dev/md0c` is out of sync with `/dev/md0b`; `/dev/md0b` is the "good" metapartition. If `/dev/md0c` is the "good" metapartition, substitute `/dev/md0c` in the following example. Note that `/dev/md0c` must be *active* before you run `metasync`; if `/dev/md0c` has not been activated with `metainit` or if it has been cleared with `metaclear`, it must be activated with `metainit` before you run `metasync`.

Note the syntax of `metasync`: name the metamirror, then the metapartition that is being synced *from*. If you name the wrong metapartition, the good metapartition will be overwritten with bad data.

```
# metasync /dev/md0a /dev/md0b
```

The `metasync` command will bring `/dev/md0c` into sync with `/dev/md0b` by copying `/dev/md0b` to `/dev/md0c`, block by block, using a fast kernel copy. Note that you can read from and write to the metamirror during the resync. You may want to run `metasync` in the background; see "Watching Progress of Resync" below.

With no argument before the name of the metapartition to be synced from, the default buffer size of 64 blocks is used. The buffer size is the size of the buffer in disk blocks (up to a maximum of 126) used for copying from the good metapartition to its mirrored pair; under some circumstances, the resync may complete faster if a larger value is used. Using this parameter allows you to set the

resync speed to your preference: the higher the number, the faster the resync; the lower the number, the more system performance is available for other users.

Watching Progress of Resync

You can watch the progress of a resync by doing a `metastat` from another shell or by doing the resync in the background and running `metastat` in the same shell; you will be told that a mirror resync is going on and you will see the number of blocks that have been copied (see the lines with arrows in the example below):

```
# metasync /dev/md0a /dev/md0b&
# metastat /dev/md0a
*****
/dev/md0a: metamirror
        first mirror partition is /dev/md0b
        second mirror partition is /dev/md0c (off-lined)
        Mirror resync going on (from first to second)  <<<----------------------
        buffer size = 32K, 79552 disk blocks have been copied.  <<<-------------
        error threshold = 10   pass = 2
        read only from the first mirror partition   <<<-------------------------
   /dev/md0b: (first)
        3 rows, total blocks 108528
        row 0:  width 1 blocks 25024    devices:        < /dev/sd0d >
        row 1:  width 1 blocks 52224    devices:        < /dev/sd0b >
        row 2:  width 1 blocks 31280    devices:        < /dev/sd0h >

   /dev/md0c: (second)
        meta-partition is off-lined   <<<---------------------------------------
        1 rows, total blocks 147400
        row 0:  width 1 blocks 147400   devices:        < /dev/xy1g >

*****
#
```

When the resync is complete a message will be displayed on your system console. If you run `metastat` again, after the resync in complete, the second metapartition will no longer be shown as "off-lined," and the message "read only from the first mirror partition" will not be displayed.

Performing Online Backups — Mirroring Active

A mirrored metapartition can be backed up (archived) without unmounting it or taking it offline. One of the mirrored metapartitions must be detached temporarily, thus losing mirroring, but it can be reattached and resynced as soon as the backup is complete — without halting the system or denying user access to the data.

There are six steps in the procedure for backing up a file system that is being mirrored; the steps listed here are described in detail in the example below.

1. Execute `sync` command three times.
2. Use `metadetach` to detach the second metapartition from the metamirror.

3. Run `fsck` against the detached metapartition.
4. Back up the data on the detached metapartition.
5. Use `metattach` to attach the detached metapartition to the metamirror.
6. Run `metasync` to sync the newly-attached metapartition to the first metapartition.

Example

Suppose you have defined the following metamirror (`/dev/md0a`), and the two metapartitions (`/dev/md0b` and `/dev/md0c`) are in sync:

```
/dev/md0a -m /dev/md0b /dev/md0c
/dev/md0b 1 1 /dev/id000c
/dev/md0c 1 1 /dev/id010c
```

1. Execute `sync` command three times; this will reduce the chance of file system inconsistencies and might speed up the `fsck`.

```
# sync
# sync
# sync
```

2. Detach the second metapartition from the metamirror.

```
# metadetach /dev/md0a /dev/md0c
```

   Reads and writes will continue to be made from and to `/dev/md0b`; this means that the two metapartitions will no longer be in sync as soon as one write or delete is made to `/dev/md0b`.

3. Run `fsck` against the second metapartition.

```
# fsck /dev/md0c
```

Because the state of the file system is inconsistent* after the `metadetach`, you must run `fsck` to ensure that the file system is intact; otherwise the system may panic and crash — if the metapartition is mounted. The backup may be faulty, in any case. If you do not have a file system mounted on the metapartition, use whatever partition integrity checks are appropriate for the application. (If you unmount the metamirror before detaching the metapartition, you do not have to run `fsck` before performing the backup.)

* There may be "dirty pages" in the buffer cache that have not yet been flushed to the disk, causing a file system inconsistency.

4. Perform backup.

   Use `dump` or whatever other backup utility you wish to use to copy the detached metapartition to tape or some other medium.

5. Re-attach the metapartition to the metamirror.

```
# metattach /dev/md0a /dev/md0c
```

   Assuming you want to resync the metapartitions, you first have to attach the second metapartition to the metamirror. Skip this step and the next one if you do not wish to resync the metapartitions.

6. Sync the second metapartition to the first metapartition.

```
# metasync /dev/md0a /dev/md0b
```

   Assuming you want to re-establish mirroring, you have to resync the second metapartition to the first. Note that the syntax of `metasync` requires that you name the metapartition that you will be resyncing *from*. Skip this step if you do not wish to resync the metapartitions.

Performing Online Backups — Mirroring Not Active

If you plan ahead, the Metadisk driver will allow you to enable mirroring or perform on-line backup very easily, without ever having to unmount file systems. The secret is to have file systems mounted on 'mirroring-ready' metapartitions, as shown in the first box, below. By setting these up in advance, you will be able to attach the second (mirroring) metapartition at a later time.

When you want to do an online backup, attach a second metapartition, sync it to the first, and do the backup as you did in the example above. The steps to be performed, once you have defined, activated, and used a metamirror with one metapartition, are:

1. Define second metapartition.
2. Activate the second metapartition.
3. Attach the second metapartition to the metamirror.
4. Sync the second metapartition to the first metapartition.
5. Execute `sync` command three times
6. Detach the second metapartition.
7. Run `fsck` on the detached metapartition.
8. Perform backup.

Example

Suppose you have defined a metamirror (`/dev/md0a`) with only one metapartition (`/dev/md0b`):

```
/dev/md0a -m /dev/md0b
/dev/md0b 1 1 /dev/id000c
```

1. Define the second metapartition, if necessary.

   If you have not already defined the second metapartition, do so now; be sure the second metapartition is the same size, or larger, than the first metapartition. In this case, both metapartitions are partition `c`, the entire disk, and they are the same type of disks, so they are the same size:

```
/dev/md0c 1 1 /dev/id010c
```

2. Activate the second metapartition, if necessary.

   If you have not already activated the second metapartition, do so now:

```
# metainit /dev/md0c
```

3. Attach the second metapartition to the metamirror.

```
# metattach /dev/md0a /dev/md0c
```

4. Sync the second metapartition to the first metapartition.

```
# metasync /dev/md0a /dev/md0b
```

   The second metapartition must be synced to the first metapartition before being detached and backed up. Do not detach the second metapartition until `metasync` is complete.

5. Run `sync` three times; this will reduce the chance of file system inconsistencies and might speed up the `fsck`.

```
# sync
# sync
# sync
```

6. Detach the second metapartition from the metamirror.

```
# metadetach /dev/md0a /dev/md0c
```

   Reads and writes will continue to be made from and to `/dev/md0b`; this means that the two metapartitions will no longer be in sync as soon as one write or delete is made to `/dev/md0b`.

7. Run `fsck` against the second metapartition.

```
# fsck /dev/md0c
```

Because the state of the file system is inconsistent* after the `metadetach`, you must run `fsck` to ensure that the file system is intact. If you did not have a file system mounted on the metapartition, use whatever partition integrity checks are appropriate for the application.

8. Perform backup.

   Use `dump` or whatever other backup utility you wish to use to copy the metapartition to tape or some other medium.

   Assuming you do not want mirroring, you are finished with this procedure; if you wish to continue mirroring, perform the last two steps in the procedure above (`metattach` and `metasync`).

4.7. Modifying a Used Metapartition

Once a metapartition has been "used" (mounted, written to, or otherwise modified), the procedures for modifying the metapartition is more complex than the procedures used for unused metapartitions. Some examples are described below.

Changing the Definition of a Used Metapartition

If you wish to change the definition of a metapartition or metamirror that has been used — that is, if a file system has been mounted on it or it has been written to — you will probably want to save the current data before making the change to the metapartition, and you will have to restore the data after making the change. (If you only want to change one of the metamirror options, you do not have to do a backup and restore; skip those steps in that case.)

This example assumes you have a file system mounted on the metapartition; substitute appropriate activities if the metapartition is being used in some other way.

1. Unmount the metamirror or metapartition.

```
# umount /dev/md0a
```

   As always, if this command produces the message "Device busy," make sure that all shells are `cd`-ed out of this file system; if you still get the busy message, daemons or remote access are still using the file system. In the extreme case, you may have to halt the system to unmount the metapartition.

2. Back up the metapartition.

   Use `dump` or `tar` or whatever backup tool is appropriate. It may be advisable to back up your data to disk, since this is much faster than backing up to tape.

---

* There may be "dirty pages" in the buffer cache that have not yet been flushed to the disk, causing a file system inconsistency.

3. Clear the metapartition.

```
# metaclear /dev/md0a
```

4. Edit the `metatab` file.

   Make whatever changes you wish to `/etc/metatab` — within the rules of the MetaDisk Driver — and save the file. You might want to increase the size of the metapartition by concatenating physical partitions together, for instance.

5. Activate the metapartition.

```
# metainit /dev/md0a
```

   Unless you have defined an illegal configuration, the metapartition will be set up; run `metastat` to confirm.

6. Run `newfs` to create a new file system on the metapartition.

   Since you have created a new metapartition, you have to create a new file system on it. Be aware that any data on the metapartition will be lost; be sure you have backed up vital data.

7. Restore the data from the dump medium.

   Use `restore` or other appropriate tool to restore the backed up data that you saved earlier to the new file system.

8. Mount the metapartition.

```
# mount /dev/md0a
```

   The example above assumes you wish to mount the metapartition on the same directory as before; modify `/etc/fstab` if you wish to mount it elsewhere.

Replacing a Disk: Concatenated Metapartition

To replace a disk which includes a physical disk partition which is a component of a concatenated metapartition (but not a metamirror component), do the following. Note that you must run `newfs` on the metapartition and restore backed-up data to it after replacing the disk; *part* of a concatenated metapartition is useless.

1. Back up the data, if possible.

   If the disk is usable, back up all needed data before continuing.

2. Halt the system.

3. Repair or replace the failed device — drive, cable, or controller.

4. Boot the system in single-user mode.

5. Edit `/etc/fstab`, commenting out the file system(s) mounted on the metapartition(s), if any.
6. Boot the system multi-user.

   Assuming `metainit -r` and `metasync -r` are in the appropriate `rc` files, you will not have to run `metainit` and `metasync` manually on the metapartitions that do not use the affected disk.
7. Clear metapartition(s) on failed disk with `metaclear`.
8. Format, partition, and label the disk, if necessary.
9. Edit `/etc/metatab` to replace the failed partition(s), if necessary.

   If you do not use the same physical partition(s) to define the affected metapartition(s), you will have to modify `/etc/metatab` to reflect the changes made.
10. Activate affected metapartition(s) with `metainit`.
11. Create file system on metapartition(s) with `newfs` (assuming this is a file-system application; otherwise, skip this step).
12. Restore the data to the metapartition(s) from the latest backup.
13. Re-activate mounting of the metapartition(s) in `/etc/fstab`.
14. Mount the file system(s).

Mirroring: Replacing a Failed Disk Partition — Spare Disk Available

If one of a pair of active mirrored metapartitions is taken offline (indicated by console messages and/or your system logging mechanism) because of a disk read or write failure, or by the command `metaoffline`, the metapartitions will be out of sync when the next write is made to the still-active metapartition.

If you have sufficient spare physical partitions, you can replace the failed partition and bring the metapartitions back into sync with no downtime. You can continue to use the good metapartition while you are replacing the failed partition and resyncing the metapartitions.

Note that the replacement partition must be at least as large as the partition that was taken offline; `metareplace` will fail if the replacement partition is smaller.

1. Format, partition, and label the replacement disk, if necessary.
2. Replace the bad partition.

   Run `metareplace`, substituting an unused physical partition for the partition that failed; the physical partition must be at least as large as the one it replaces. Suppose you have defined the following metamirror in `/etc/metatab`:

```
/dev/md0a -m  /dev/md0b /dev/md0c
/dev/md0b 2 1 /dev/id000c 1 /dev/id001c
/dev/md0c 2 1 /dev/id004c 1 /dev/id005c
```

If physical partition `/dev/id004c` fails (indicated by error messages on the console or by an error flag on the device in `dmt`), you can replace it with a spare disk partition, e.g., `/dev/id006c`. Note that you name the metamirror (`/dev/md0a`) — *not* the metapartition that has a bad partition.

```
# metareplace /dev/md0a /dev/id004c /dev/id006c
```

This will detach `/dev/id004c` from metapartition `/dev/md0c`, replace it with `/dev/id006c`, and modify `/etc/metatab`. While this is going on, you can continue to read from and write to `/dev/md0a`, using the good metapartition, `/dev/md0b`.

3. Sync the metapartitions.

   Run `metasync` to resync the mirrored metapartitions, naming the metamirror (`/dev/md0a`) and the metapartition from which to do the syncing (`/dev/md0b`):

```
# metasync /dev/md0a /dev/md0b
```

This will sync `/dev/md0c` to `/dev/md0b` dynamically, allowing you to continue to write to `/dev/md0b` while the resyncing process is going on. See "Syncing Metamirror Metapartitions" above for more information about syncing.

Mirroring: Replacing a Failed Disk Partition — No Spare Disk Available

If one of a pair of active mirrored metapartitions is taken offline (indicated by console messages and/or your system logging mechanism) because of a disk read or write failure, or by the command `metaoffline`, the metapartitions will be out of sync with the next write to the still-active metapartition. If you don't have a spare disk partition available, do the following to replace the failed disk. If you can do without mirroring for a time, you can wait until it is convenient to replace the failed disk; you will be without mirroring until you replace the disk and run `metasync`, of course.

1. Halt the system. You might want to use `metahalt`, to prevent automatic running of `metasync -r` during reboot.
2. Repair or replace the failed disk drive.
3. Boot the system.

   If `metainit -r` and `metasync -r` are in the appropriate `rc` files, all metapartitions and metamirrors will be activated, and all mirrored metapartitions will be synced, unless one of the metapartitions is offline, as will be the case for the metapartition with the failed disk partition.

4. Format, partition, and label the disk, if necessary.
5. Sync the metapartitions.

   Run `metasync` to resync the mirrored metapartitions, naming the metamirror (`/dev/md0a`) and the metapartition from which to do the syncing (`/dev/md0b`):

```
# metasync /dev/md0a /dev/md0b
```

This will sync `/dev/md0c` to `/dev/md0b` dynamically, allowing you to continue to write to `/dev/md0b` while the resyncing process is going on. See "Syncing Metamirror Metapartitions" above for more information about syncing.

4.8. Rebooting Without Resyncing

If you have inserted `metasync -r` in `/etc/rc.local`, a system halt and reboot will automatically cause resyncing, since there is no way to be sure a pair of mirrored metapartitions are in sync after a system halt. If you need to do a planned halt or reboot, you can use one of the MetaDisk commands listed below to halt and/or reboot. These commands set a MetaDisk flag before halting and/or booting, telling the MetaDisk Driver that no resync is necessary, thus preventing a time-consuming resync.

- metahalt
- metafasthalt
- metareboot
- metafastboot

Use one of these commands instead of their UNIX equivalents if you have any mirrored metapartitions in use. When you boot the system, `metasync` will note the flag and will skip the resync.

4.9. Safety Tips

The MetaDisk Driver's mirroring capability provides file insurance, for the greatest safety possible in a UNIX system. This section gives a few tips to ensure that you'll actually realize that safety. Many of these tips are included in "Cautions and Admonitions" earlier in this chapter.

Whenever concatenating, use mirroring as well, if you have the spare disk capacity. Concatenating partitions reduces effective reliability somewhat, so mirroring any critical data stored in a concatenated metapartition is a good idea.

When mirroring, keep the two metapartitions on separate disks. Data protection is diminished considerably if both metapartitions are on the same disk.

When mirroring, keep the metapartitions on different controllers, if possible. Controllers and associated cables may fail more often than disks, so organize your metamirrors so the two copies are as independent as possible. This also helps performance.

If the MetaDisk Driver takes a metapartition off-line, consider unmounting other file systems mounted on the disk where the failure occurred. Since each disk partition is independent, up to seven file systems may be mounted on a single disk. If the MetaDisk Driver has "trapped" a failure (as indicated by `metastat`), it is likely that other partitions on the same disk may experience failures soon, also. File systems mounted directly on disk partitions do not have the protection of MetaDisk Driver error handling, and leaving such file systems mounted can leave you vulnerable to crashing the system.

Minimize the amount of time you run without mirroring. During resyncing and on-line backup intervals, you do not have the protection of mirroring. Your system administrator should run `metastat` regularly on any servers using the MetaDisk Driver.

Use the IOadmin tools `fsm` and `dmt` to help you understand your MetaDisk configuration *before* you activate the metapartitions. These tools are meant to help prevent human error; use them. Use the `df` command to double-check the current file system activity and configuration before activating metapartitions.

Do not mount file systems on a meta-component. If a physical partition is participating in a metapartition of any kind, you must not mount that partition as a file system. Unmount any physical partition you intend to use as a metapartition *before* you activate it.

4.10. How the MetaDisk Handles Errors

The Metadisk has three operational modes — mirroring, mirror-resyncing, and concatenation. Each of these modes handles errors differently, so it is important to understand the underpinnings of the meta-devices before going into details.

A metapartition operates at the "logical block" level, so it doesn't get involved with physical blocks. This means that the physical disk controller and host adaptor software handles many errors before they are passed on to the MetaDisk Driver. For example, if a disk drive encounters a bad block on reading or writing, the disk driver will re-try the operation several times. If the block fails repeatedly, most of disk drivers automatically re-map the bad block to a good one in real time. Older drivers require that you do this manually (using the `format` utility). Since the physical disk driver handles bad blocks at the physical level (and hands off an OK logical block to the MetaDisk Driver), the MetaDisk Driver never has to deal with this issue.

However, if the re-mapping of blocks fails (because there's a fatal head crash, or a mechanism or electronics failure), an error message is passed from the physical driver to the MetaDisk Driver — and that's where the MetaDisk Driver operating mode becomes important.

In mirroring mode, if there is a read error, the MetaDisk Driver will re-try the I/O until it has exceeded the error threshold (set in `/etc/metatab`); if there is a write error, the MetaDisk Driver will not retry. Once the MetaDisk Driver has given up on the I/O operation, it takes the failed metapartition offline. This means that the MetaDisk Driver will no longer try to use that metapartition — but other parts of the system (e.g., file systems) may still try to use the disk where the failure occurred.

When a metapartition has been taken offline, you might want to consider taking the disk where the failure occurred out of service — by unmounting any file systems mounted on the drive. Run `metastat` to see the current configuration and status of all metapartitions in a system, noting especially any that are offline.

Note that anyone using the metamirror has not had to worry about the failed metapartition: users continue working normally with the surviving copy. But now they are running without the protection of mirroring. Any hard disk failure encountered at this time will be passed directly through the MetaDisk Driver, up

to the file system and the user's applications.

Resyncing of the failed metamirror resolves this issue by re-establishing mirroring. Until the resync is complete, however, users are vulnerable to a disk failure. If the surviving copy of the data now encounters a disk failure, the MetaDisk Driver can only pass the error up to the users' applications.

If a disk error occurs on the "embryonic" copy that's being created, users aren't impacted: the resync process will stop, but the users keep on running. Under this circumstance, the resync process can be restarted once the failed disk is replaced.

For concatenated metapartitions, error handling is straightforward: any disk crash involving a disk which is a component of a metapartition will cause the entire concatenation to fail. Once this occurs, none of the data stored within the concatenation can be used; you will have to get your backup tapes and do a restore. Consequently, we highly recommend that any concatenation be mirrored, for safety and better performance.

4.11. Performance Tips

The metadisk is not intended to improve disk performance; however, in some cases it can do so. Even more important, a poorly-configured MetaDisk can hurt performance. This section gives a few tips to increase the probability that you'll actually realize the best performance possible.

When concatenating, don't concatenate partitions on a single drive. For example, don't concatenate `/dev/sd1e` and `/dev/sd1h`; you can do so, but it will hurt performance. If you want to concatenate these partitions together, try make sure they are far apart in the list of partitions in the metatab. For example:

```
/dev/md1a 4 1 /dev/sd1e 1 /dev/sd2a 1 /dev/sd3a 1 /dev/sd1h
```

will tend to have better performance than:

```
/dev/md1a 4 1 /dev/sd1e 1 /dev/sd1h 1 /dev/sd2a 1 /dev/sd3a
```

due to head-movement times.

In mirroring, don't have both copies of the data on the same drive. Always try to use two drives in mirroring, since two writes to the same disk will always have to be serial writes.

Try to have all drives in a metapartition (metamirror, or concatenation) on separate data paths. For SCSI drives, this means separate host adaptors. For SMD and IPI drives, this means separate controllers. By having the metamirror or concatenation spread the I/O load over several controllers, there's a better chance of good performance.

Use `pmt` to identify heavily-loaded disks and controllers. By examining disk transfers per second, up to 10 spindles can be monitored simultaneously (using the bar-chart mode). Analyzing historical data (a month's data or more), you can identify which disks are most heavily loaded — and using `dmt`, you can determine which controllers are most heavily loaded. With this information, you can identify metapartitions that could benefit from the read-mode options discussed below, or from re-configuration.

In a metamirror, experiment with the read options of the MetaDisk Driver. The default mode is to alternate reads in a "ping-pong" fashion between the two disks. This is the default because it tends to work best for multi-user, multi-process activity with the UNIX file system.

However, if one of the disks or controllers participating in a metamirror is heavily loaded by other activity (e.g., another file system on another of its partitions), you may want to use the `-r` option. This mode will force the MetaDisk Driver to read only from one partition, thus avoiding further overloading the other partition. (Note that the MetaDisk Driver does not automatically choose which partition in overloaded; you must specify this by putting the more heavily-loaded metapartition as the last one in the line that describes this metamirror in the metatab. See Chapter 3 for details of how to use this option.)

In some circumstances, the `-g` option can improve performance by minimizing head motion and access times. This option is most effective when there is only one partition per disk, only one process at a time using the partition/file system, and when I/O patterns are highly sequential. See Chapter 3 for details of how to use this option.

Don't mix disks of different technologies in a single metapartition or metamirror. This will tend to make performance the "lowest common denominator" of all the devices in the metapartition.

Don't mix disks or controllers of widely varying performance in a single metapartition or metamirror. Particularly in old SCSI or SMD storage devices, different models or brands of disk or controller can have widely varying performance. Mixing the different performance levels in a single metapartition or metamirror can cause performance to degrade significantly.

5

IOadmin Overview

IOadmin is comprised of three tools — Device Management, File System Management, and Performance Monitor — which help you monitor and administer your Sun systems. Each tool is described in detail in subsequent chapters; this chapter defines some general terms and describes the user interface for the tools. Note the following prerequisites for running the tools:

- You must be running under SunView on your IO Manager machine. (Note, however, that you can monitor a server remotely, even if it is not running SunView.)
- The IOadmin daemon (`rpc.ioadmd`) must be running on the server.

Although not required, you will probably also want to include the IOadmin binary directory (the directory where the IOadmin tools are installed) in your command path.

5.1. About IOadmin Directories

During installation, three primary IOadmin directories are created: the tools directory (`ioadm/bin`), the daemon directory (`ioadm/etc`), and the man page directory (`ioadm/man`). As noted in Chapter 2, you can move the man page directory to a central man page location if you like. Note the following about the other two directories.

- **tools directory:** The directory where the IOadmin tools (`dmt`, `fsm`, and `pmt`) reside; by default, the directory is `/usr/ioadm/bin`, but you can choose a different location during installation, or you can move the tools directory later.
- **daemon directory:** The directory where the IOadmin RPC daemon, Performance Monitor daemon, and other IOadmin utilities reside; by default, the directory is `/usr/ioadm/etc`, but you can choose a different location during installation, or you can move the directory later.

5.2. Windows: The SunView Environment

IOadmin is a package of SunView window tools, which means that they run only in the SunView environment. Once you have started SunView and opened up a ShellTool or CommandTool window, you can start up one of the IOadmin tools.

Primary IOadmin Tool Windows

When you start one of the IOadmin tools, the *primary window* for the tool is displayed, usually comprised of two or more subwindows, each with a particular purpose, and including functional *buttons* for performing actions or displaying *menus*. The figure below shows you the various parts of an IOadmin primary window — the Performance Monitor startup window, in this case.

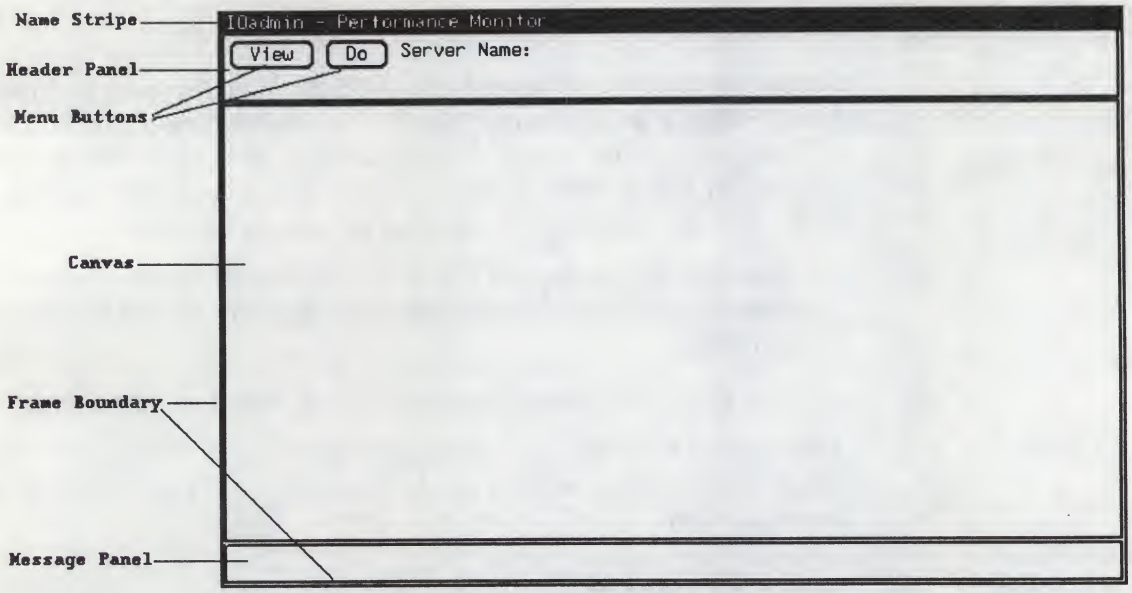


Figure 5-1 *Elements of an IOadmin Window*

Name Stripe: the name stripe displays the name of the tool you are running.

Header Panel: the header panel is the subwindow where you designate which server you wish to administer (by typing its name), and what you wish to view or do by selecting items from the View or Do menus. Note that once you have viewed a server, its name will be displayed in a popup menu if you press RIGHT in the Server Name area; move the mouse to select a server, and release the mouse button to display information about the server.

View and Do Menu Buttons: the View and Do buttons in the header panel are menu buttons: if you move the mouse pointer to one of the buttons and press and hold the RIGHT mouse button, the menu will be displayed. Move the mouse until your choice is highlighted and release the mouse button to select that item.

Canvas: the canvas is the main display area for the IOadmin tools. This is where you will see the graphical representation of hardware configurations, file system layouts, and performance monitoring charts.

Frame Boundary: the frame boundary, which completely surrounds the window, serves the normal SunView functions of moving, closing, and resizing windows, bringing windows to the foreground or putting them in the background, and quitting the current tool.

Message Panel: the message panel displays informative or error messages from the tool you are running. Informative messages (such as "Press RIGHT on View button for System Map") help you figure out what to do next. Error messages, which are accompanied by an audible beep and a visible flash of the screen, tell you when you have forgotten to do something (such as saving some text changes) or when there is a problem with the tool or the system.

Popup Windows

At various times when you select a View or Do menu item, a new window will "pop up," usually near the top-right corner of the primary window. These popup windows contain detailed information about the item selected in the View menu, or give you the opportunity to provide more information about a function you selected from the Do menu.

Text Subwindows

Some popup windows (such as the Note Pad in Device Management) are text subwindows, which means that you can add, move, delete, copy, and generally manipulate text in these subwindows as you can in any SunView text window.

Dismissing Popup Windows

Many popup windows include a Dismiss button, on which you can click LEFT to get rid of the window. Text subwindows usually include a Save button, in addition, for saving any changes you have made; if you try to dismiss the window without saving the changes, you will be warned and given a chance to save the changes before dismissing the window.

Note that you can also dismiss a popup window by selecting "Dismiss" from the popup window which is displayed if you press RIGHT on any border of the IOadmin popup. Other choices are "Move," "Resize," "Front," "Back," and "Redisplay."

Scrollable Canvases and Subwindows

All of the IOadmin tools have *scrollable* canvases or subwindows — if scrolling is needed to display all of the information that is available.

File System Management has vertical scrollbars; Performance Monitor has horizontal scroll bars. Device Management has vertical and/or horizontal scrollbars, depending on whether extra space is needed in the vertical or horizontal direction. Most popup windows in all of the tools have scrollbars, also.

Use the scrollbars just as you do in other SunView applications: click or press LEFT on the scroll arrows to move the contents UP, click or press RIGHT to move the contents down, and click MIDDLE to center the contents around the relative position of the pointer in the gray portion of the scrollbar.

Moving a Window

Move an IOadmin window just as you would any SunView window — by moving the pointer to any frame border, pressing the MIDDLE mouse button, and moving the mouse, or by displaying the frame menu by pressing RIGHT anywhere on the frame border, selecting Move from the frame menu, and following the instructions in the message box.

Resizing a Window

Resize an IOadmin window by displaying the frame menu, selecting Resize, and following the instructions; or hold down the Control key, move the pointer to a frame border, and press MIDDLE while moving the mouse. To *zoom* an IOadmin window to its maximum vertical screen size, move the pointer to the frame of the primary window, hold down the CTRL button, and click LEFT. Repeat the process to *unzoom* the window.

IOadmin windows are resized logically, according to which of the subwindows should absorb or accept the resizing: for all of the IOadmin tools, the *canvas* takes all vertical resizing; that is, the canvas gets taller or shorter during resizing while the header panel and message panel stay the same height.

Printing a SunView Screen

If you want to print an image of a SunView screen, use the UNIX utility `screendump`. You could create an alias such as the following in your `.cshrc` file to make it easy to remember how to print the image. Remember to `source` the `.cshrc` file after saving it with the alias defined.

```
alias sdump 'screendump | pssun -S 8 | lpr'
```

If you have a Sun-4 machine and you include `-r` (which rotates the image) after `pssun` in the alias shown above, the screendump will probably fail. You may have to use some simpler mechanism for printing the screendump, such as the following:

```
alias sdump 'screendump | lpr -v'
```

5.3. Buttons and Menus

Two types of buttons are discussed in this section: the physical buttons on your system "mouse" and graphic buttons in various IOadmin tool windows. There are various graphic menus in the tool windows, also.

Mouse Buttons

The terms "LEFT," "MIDDLE," "RIGHT," "Click," and "Press" refer to the three mouse buttons and have specific meanings within this document:

- **LEFT:** The left mouse button. Used for selecting from a list or activating a function by clicking on a window button.
- **MIDDLE:** The middle mouse button. Primarily used for moving or resizing windows. Also used for multiple selections within FSM.

- **RIGHT:** The right mouse button. Primarily used for displaying popup menus.
- **Click:** Depressing a mouse button and releasing it quickly. The LEFT mouse button is primarily *clicked*. To open a closed IOadmin tool, for instance, *click LEFT* on the tool icon. To get rid of a popup window, *click LEFT* on the Dismiss button.
- **Press:** Pressing and holding a mouse button down. The MIDDLE and RIGHT mouse buttons are usually *pressed*. To display the View menu in Device Management, *Press RIGHT* with the pointer in the canvas. *Press RIGHT* anywhere in the border of the tool window to display the standard Close, Open, Resize, etc., menu.

View and Do Menus

The View and Do menus, which pop up when you press RIGHT on the [View] and [Do] buttons, are common to all IOadmin tools. Select [View] to display information and [Do] to perform various functions.

Select an item from a menu just as you do in other SunView tools: move the mouse until the item you wish to select is highlighted, and release the mouse button. If an item is available for selection, it will be in bold type; if it is unavailable, it will be in a lighter, gray type face, and will not be highlighted by moving the pointer over it.

Do Menu: Need for root Password

Some functions in the Do menus of `fsm` and `dmt` require that you become the superuser. For such functions, a window will pop up, asking you to type in the root password; once you type the password and press [Return], the function will be performed. If you start the IOadmin tool as root on the server, you will not be asked to enter the password. Similarly, if your machine name and user ID are included in the `/.rhosts` file on a server, you will be able to perform these functions without entering the root password. Once you have entered the root password for a server, you will not have to re-enter it unless you change servers or start a new tool.

Server Name Menu

The Server Name area of the header panel contains a pulldown menu of available servers. Press RIGHT to see which servers are available. There are three ways to

insert server names in this menu; see "Changing Servers" below for instructions.

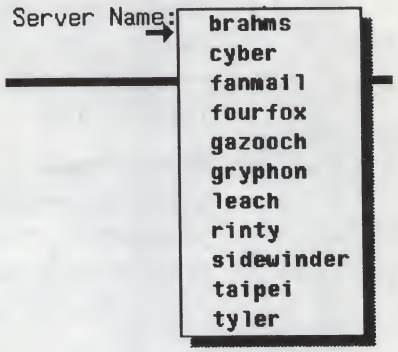


Figure 5-2 *Server Name Menu*

Cycle Buttons

At various times you will be asked to select from a group of choices arrayed under a *cycle button*, characterized by two half-circles with arrows:

If you click LEFT in the area of the button, you will cycle through the choices, one at a time; stop when the choice you want is displayed. If you press RIGHT in the area of the cycle button, a menu of the choices will be displayed; move the mouse to select the one you wish and release the mouse button to select that choice.

5.4. Starting IOadmin Tools

To start an IOadmin tool, type the name of the command and press Return; you must be running SunView before you start the tool. To start Device Management from the command line, for example, type

```
% dmt
```

and press Return, adding the ampersand (&) to run the tool in the background. This is all you have to type if you are in the IOadmin tools directory (`/usr/ioadm/bin` by default), or if you have set up your command path so that the IOadmin tools directory is included. Otherwise, include the full path name:

```
% /usr/ioadm/bin/dmt&  (substitute tool location if different)
```

Command Line Options

You can start the IOadmin tools with special SunView characteristics, as described in the *SunView User's Guide*; you can designate the color of the windows, for instance, or the font to be used for text.

In addition, there are three IOadmin options which can be added to the command-line startup (include a space between the option and its argument, if any):

-h [*servername1 servername2 servername3 etc.]*: start the tool with the named servers preselected when the tool comes up; each server will appear in the Server Name pulldown menu.

-m: run the tool in monochrome on a color monitor.

-t *seconds*: change the default timing interval from 300 seconds to another value. This is the interval for updating dynamic displays for Device Management and File System Management.

Starting All Three Tools at One Time

If you wish to start all three IOadmin tools at one time, insert an alias like the following in your `.cshrc` file (or appropriate file if using a different shell). The following example assumes the IOadmin tools are in the default location, `/usr/ioadm/bin`; substitute the proper paths if you have installed them elsewhere.

```
alias ioadmin '/usr/ioadm/bin/dmt& /usr/ioadm/bin/fsm& /usr/ioadm/bin/pmt&'
```

Then `source` the `.cshrc` file to activate the alias; type `ioadmin` to start the tools.

```
% source ~/.cshrc
% ioadmin
```

All three tools will start up, with all of the servers in "Specifying Servers to Be Monitored" below for instructions).

Specifying Servers to Be Monitored

To pre-specify which servers you wish to monitor and administer whenever the IOadmin tools are started up, create the file `separate line:`

```
brahms
cyber
fanmail
gazooch
gryphon
leach
rinty
taipei
tyler
```

All of the servers listed will be available in the Server Name menu when the tool is started up.

5.5. Including Devices in ioadmd.conf

The file `ioadmd.conf`, which should be created in the same directory as the IOadmin daemon, `rpc.ioadmd`, serves three purposes:

- prevents inadvertent tape rewinds and CD-ROM disc ejections
- causes `fsm` to ignore certain devices
- speeds up `fsm`

The procedure is the same, for all three purposes:

1. Include the device name in `ioadmd.conf`
2. Kill the IOadmin daemon (`rpc.ioadmd`) and restart it.

Read all of this section before editing `ioadmd.conf`. Do the following to kill the IOadmin daemon and restart it *after* modifying `ioadmd.conf`:

```
# ps aux | grep rpc.ioadmd
root  3440  0.0  2.3  104  176 ? I 10:41  0:05 rpc.ioadmd
# kill 3440        (Substitute actual process number)
# rpc.ioadmd
#
```

Contents of `ioadmd.conf`

The IOadmin daemon configuration file, `ioadmd.conf`, contains the names, one per line, of the devices which `dmt` and `fsm` will treat in a special way. Entries in the file may be two-character, device name prefixes (`st, sd`), partial device names (`st0, sd1, sd1a`), or full names of no-rewind tape drives (`nrst0, nrst1`). See "Preventing Unwanted Side Effects" and "Speeding Up `fsm`" below for specific examples.

Preventing Unwanted Side Effects

Tape drives queried without the no-rewind option and CD-ROMs in audio mode may act inappropriately when queried by `fsm` or `dmt`: a tape in a drive may rewind, and a disc in a CD-ROM may eject. To prevent such occurrences, include the two-character device name for the devices in `ioadmd.conf`. In the following example, SCSI tape drives (`st`) and CD-ROM drives (`sr`) are excluded:

```
#   Include comments following pound sign (#).
st  # SCSI tape drive
sr  # CD-ROM drive
```

Speeding Up fsm

To speed up `fsm`, you can remove non-existent block devices from IOadmin probing by including the device names in the file, `ioadmd.conf`. You could also remove the devices from `/dev`, but modifying `ioadmd.conf` is a less drastic solution.

Why fsm Is Slowed Down

The `rpc.ioadmd` daemon builds a list of the active disk partitions during each `fsm` update operation. It does this by searching through the `/dev` directory for block devices (disk drives and some tape drives) and "touching" them to get the status. Trying to touch a non-existent SCSI device takes a relatively long time due to a timeout delay. If the `/dev` directory contains entries for several SCSI devices (e.g., eight entries each for `sd0` through `sd6`) which do not exist, it can take up to 30 seconds to try to touch these devices.

And this delay will occur every time you select a new server or whenever the display is updated (every five minutes, by default). In addition, the delay during a tape rewind operation can be two minutes or longer: if a redisplay command happens to occur while a tape is rewinding, the redisplay will not be completed until the tape is fully rewound and ready to be opened.

Preventing Undue Delays

One method to deal with this delay is to delete the extraneous sd entries in the `/dev` directory so that the `rpc.ioadmd` daemon cannot find them. Alternatively, the `ioadmd.conf` file can be used to force the `rpc.ioadmd` daemon to ignore these extraneous devices. For example, to force `fsm` to ignore all SCSI disk devices on a server (e.g., a Sun-4/490 which has only IPI drives) you could create an `ioadmd.conf` file with the following entries:

```
st
xt
mt
fd
sd
```

The `sd` entry would force `rpc.ioadmd` to ignore all SCSI disk devices when doing an update for `fsm`. This entry would cause `dmt` to assume that the `sd` devices were tapes; `dmt` would then try to touch the devices; the touch operations would fail and `dmt` would not display the devices on the system map.

On the other hand, if the server had some SCSI disks (e.g., `sd0`, `sd2`, and `sd6`) the `ioadmd.conf` file could contain:

```
st
xt
mt
fd
sd1
sd3
sd4
sd5
```

With this list `fsm` and `dmt` would still recognize the `sd0`, `sd2`, and `sd6` devices, but would ignore the `sd1`, `sd3`, `sd4`, and `sd5` devices, thus providing faster update response.

How ioadmd.conf Works

When the IOadmin daemon, `rpc.ioadmd`, is started up, it looks for `ioadmd.conf` in its home directory; if `ioadmd.conf` is found, `rpc.ioadmd` reads it and stores the list of device names. If `ioadmd.conf` is not found, `rpc.ioadmd` uses a default internal list, which includes `mt`, `xt`, `st`, and `fd`.

Both `dmt` and `fsm` determine the server status by asking the `rpc.ioadmd` daemon to "touch" the device files in the `/dev` directory. The touch usually consists of opening the device and interrogating the kernel device drivers for status information. `dmt` only touches devices which are disks, or tapes with controllers; `fsm` only touches block devices — disks and some tape devices. The two tools, `dmt` and `fsm`, interpret the entries in the list differently:

- `dmt` uses the list to match device names which it interprets as tape devices; it touches the devices listed, prepending `nr` (no rewind) to the names in the list. If the name resulting from prepending `nr` to a name in the list is a legitimate tape device name, such as `nrst0`, `dmt` will touch the device and display the tape icon, with `on` or `off` on the icon, indicating the drive's status. On the other hand, if a *full* device name, such as `nrst0`, is found in the file, `dmt` will assume the device is a tape drive, but will not touch it and will show no status indicator on the icon. No "Information" will be available for such devices.
- `fsm` uses the list to indicate which block devices it should not touch. This is specifically needed for tape devices which can appear as block devices (some 1/2" tape devices — `mt` and `xt`), and which will rewind if touched by `fsm`. `fsm` will match the first portion of the device names; if ioadmd.conf contained:

  ```
  mt
  fd
  ```

  then `fsm` would not touch any block devices that begin with `mt` or `fd`, (e.g., `mt0`, `mt4`, `mt8`, `mt12`, `fd0`, `fd0a`, `fd0b` and `fd0c`).

Sample ioadmd.conf File

Note: in the examples below the server is assumed to have two SCSI tape devices, `st` and `st1`. If an `ioadmd.conf` file is created which contains just the device name:

```
st
```

the `rpc.ioadmd` daemon will match device names like `st0` and `st1` with `st` and indicate that these devices are tapes. `dmt`, reading these tape device names, will touch these devices by opening `/dev/nrst0` and `/dev/nrst1`, respectively.

If the `ioadmd.conf` file contains the names

```
st
nrst0
```

the daemon will match the names `st0` and `st1` with `st`, but will also exactly match the name `nrst0`. The daemon will indicate to `dmt` that both devices (`st0` and `st1`) are tapes, but the `st0` device should not be touched while the `st1` device will be touched. The full specification of the no-rewind, raw device name must be used to get the no touch behavior ; for example, if `nsrt` was used instead of `nrst0` in the file, `dmt` would not match any no-rewind, raw tape devices and `dmt` would touch `st0` and `st1` devices.

If the `ioadmd.conf` file is empty or does not contain the name of a tape device which exists, `dmt` will get an indication that the device is a disk and attempt to touch it like a disk. The touch operation will fail; `dmt` will ignore the tape device(s) and refuse to display them on the system map.

Shown below is a sample of a complete `ioadmd.conf` file, with notes explaining the purpose of each entry.

```
#   Include comments following pound sign (#).
fd      # floppy disk drive
mt      # Master tape
st      # SCSI tape drive
sr      # CD-ROM
xt      # Xylogics tape drive
nrst0   # dmt:  Do not try to open this SCSI tape drive
nrst1   # dmt:  Do not try to open this SCSI tape drive
```

5.6. Changing Servers

There are three ways to specify server names for the IOadmin tools:

- Type the name of a server while the tool is running — with the cursor in the Server Name window.
- Include the server names on the command line when starting the tool — following `-h`:

```
# fsm -h brahms cyber gazooch rinty
```

- Include the server names in the file `.ioadminhosts` in your login directory.

Any server specified in any of the three ways listed above will appear in the pulldown menu in the header panel (up to a limit of 105 servers). To change servers, move the pointer to the area to the right of Server Name, press RIGHT to display the menu, move the mouse until the server you want is selected, and release the mouse button.

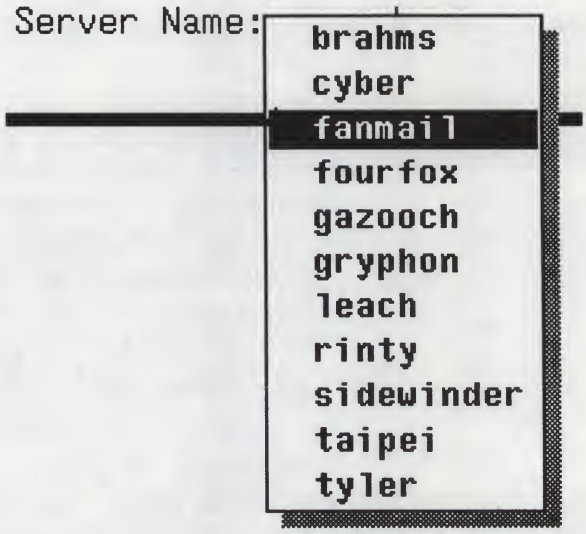


Figure 5-3 *Server Name Menu: Server Selected*

If you wish to look at a server that is not in the pulldown menu, do the following:

1. **Move the pointer to the header panel.**

   This activates the text cursor, which is probably sitting at the end of the server name.

2. **Delete the current server name**

   Delete in your normal way (with the Delete or BackSpace key, or with Control-u, etc.).

3. **Type in the new server name and press Return.**

   You will see a message in the message panel telling you that the tool is trying to access the server and asking that you please wait.

Assuming the network is working properly and `rpc.ioadmd` is running on the server, there will be a slight delay and the display will change to show you information about the new server. Read the server's console messages if there is a problem.

5.7. Tool Icons: Closing and Opening

When you are temporarily finished using an IOadmin tool, you can close the tool to icon form; you can then quickly use the tool again by opening the icon.

Closing a Tool to an Icon

Close an IOadmin tool to an icon in the normal SunView way: press RIGHT anywhere in the frame border to display the frame menu, select Close, and release the mouse button. Note that the name of the server being viewed is on the icon. The three icons, from left to right, are Device Management, File System Management, and Performance Monitor.

Figure 5-4 *IOadmin Tools Icons*

Opening a Tool from an Icon

Open an IOadmin tool from its icon by clicking LEFT on the icon.

5.8. Quitting IOadmin Tools

Quit the IOadmin tools just as you do other SunView tools: press RIGHT anywhere in the frame border and select Quit from the frame menu.

5.9. Log Files

Both `dmt` and `pmt` (`pmlogd`) create log files as they run; you may want to delete them periodically. Files written by the tools reside in `/var/spool/ioadm` on the server, and are named as follows:

dmt:

for Note Pad:

server.scrpad
<device name>.scrpad

for disktest:

log.disktest.<machine timestamp>

for filetest:

log.filetest.<machine timestamp>

for log files:

emerg.ar
emerg.log
notice.log
alert.log
crit.log
err.log
info.log
warning.log

pmt:

for log files:

pm<timestamp>

6

Device Management

Device Management is a tool to help you manage your mass storage devices — servers, controllers, disk drives, tape drives, and attached free devices. Device Management is primarily a monitoring tool, providing easy access to vital information about each of the devices you manage, but it also includes disk and file system testing functions.

6.1. Starting Device Management

Start Device Management from the command line as described in Chapter 5: type `dmt` and press Return:

```
% dmt &
```

If you have entered server names in `.ioadminhosts` in your login directory, the name of the first server in the list will be displayed in the Server Name field; otherwise your machine name will be displayed. If you start `dmt` with the `-h` flag, followed by a number of hostnames, the first name following the `-h` will be in the Server Name field. Press RIGHT in the area of the Server Name field to see other available servers.

You can start the IOadmin tools with special SunView characteristics, as described in the *SunView User's Guide*; you can designate the color of the windows, for instance, or the font to be used for text.

In addition, there are three IOadmin options which can be added to the command-line startup (include a space between the option and its argument, if any):

-h [*servername1 servername2 servername3 etc.*]: start the tool with the named servers preselected when the tool comes up; each server will appear in the Server Name pulldown menu.

-m: run the tool in monochrome on a color monitor.

-t *seconds*: change the default timing interval from 300 seconds to another value. This is the interval for updating dynamic displays for Device Management and File System Management.

Unless you have specified a server or multiple servers by using the `-h` option, the local Server Name will be displayed (see Chapter 5 or the man page for `dmt` for details).

If you you have not set your path variable to include IOadmin tools and you are not in the IOadmin tools directory, include the full path name.

The Device Management Window

After you have started Device Management, the primary Device Management window will be displayed (the local Server Name included if you started Device Management from the command line without the `-h` option).

6.2. Closing or Quitting Device Management

Close Device Management to an icon or quit Device Management as you do all IOadmin tools.

Closing Device Management to an Icon

Press RIGHT anywhere in the frame border and select Close; Device Management will be closed to an icon on the desktop, with the name of the current server displayed.

Figure 6-1 *Device Management Icon*

Quitting Device Management

Press RIGHT anywhere in the frame border and select Quit. You will be asked if you really want to quit; click LEFT on the Confirm button to quit, or LEFT on the Cancel button if you don't want to quit. When you click LEFT on Confirm, the primary tool window and all subwindows will disappear.

6.3. The System Map

The System Map, which is the starting point for all Device Management activities, is a graphic representation of the hardware configuration of the currently-selected server. A simple example of a System Map is shown below: the server, a Xylogics controller (*xyc0*) with one Xylogics disk drive (*xy0*), and a SCSI controller (*si0*) with a SCSI disk drive (*sd0*) and a SCSI tape drive (*st0*).

Displaying the System Map

Press RIGHT on the [View] button and select System Map (the only available choice at startup) from the View menu to display the System Map of the designated server (or the local server if no server name is included in the header panel). You can also display the System Map by moving the pointer to the header panel and pressing [Return].

The pointer will turn to an hourglass while information is gathered, and then the System Map for the current server will be displayed (*opusone* in this case).

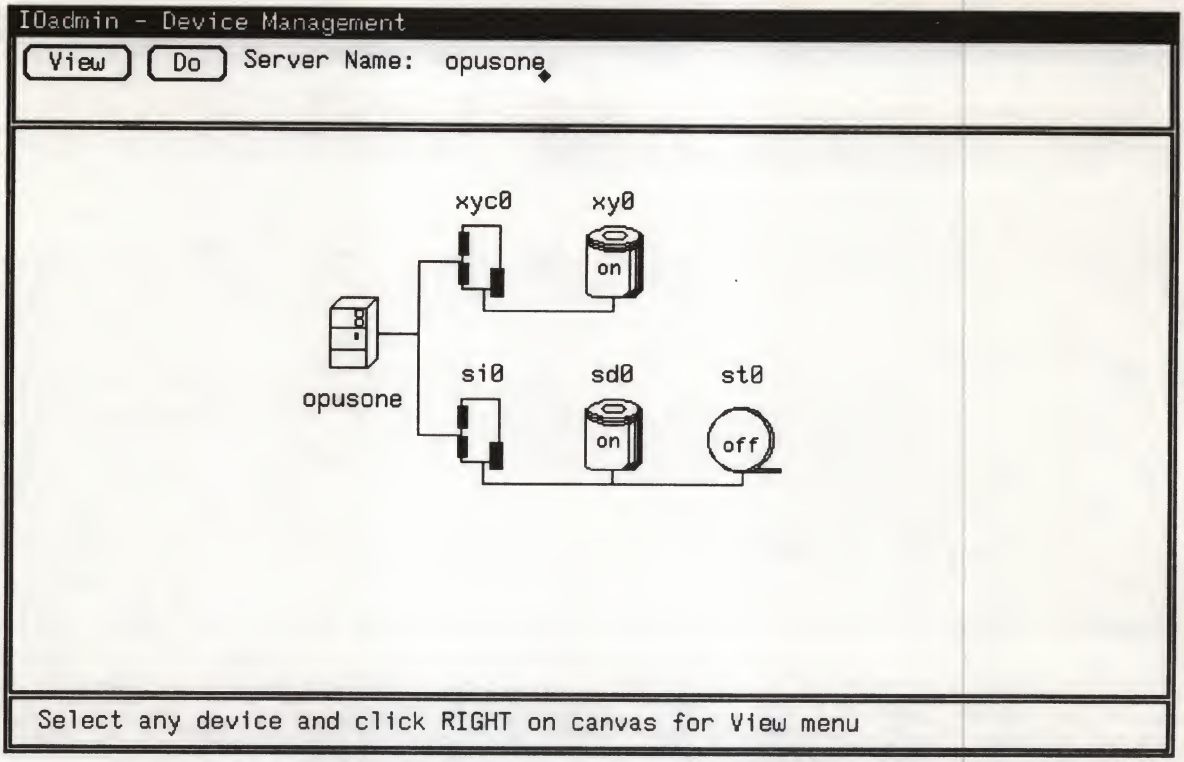


Figure 6-2 *System Map*

Selecting a Device in the System Map

Move the pointer to any device and click LEFT to select it; the device (the server, in this example) will be highlighted. Depending on what kind of a device you select, various View menu items will be available, as described below. The first device shown in a row of devices is generally a controller (`idc0, idc1, sm0` in example below). Note that a device starting with `isc` on systems with IPI disks is not a physical device, but is rather a "logical device," representing a

string controller; no information will be available for such a device.

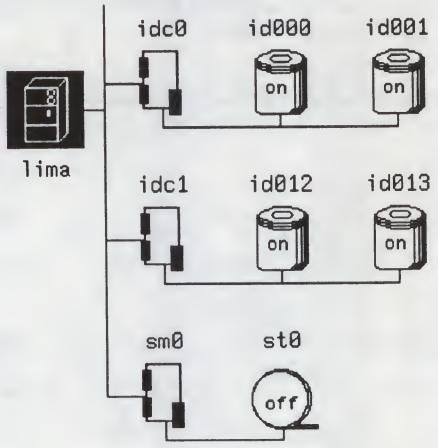


Figure 6-3 *System Map: Server Selected*

Dynamic Hardware Status Indicators

Note the "on" and "off" indications on the disk and tape drives. Since Device Management gets its status information from the system kernel, this is the true status of the devices — within the five-minute interval between updates. Any time you change servers or press [Return] with the pointer in the header panel, the kernel is queried and the status is updated.

Note: The notation "on" for a tape drive means that there is a tape in the drive and it is ready to be written to; "off" means there is no tape in the drive. If neither "on" nor "off" is shown, the tape drive is not being monitored for status; see "Including Devices in ioadmd.conf" in Chapter 5 for details.

6.4. The View Menu

Press RIGHT on the [View] button to display the View menu; you can display the View menu by pressing RIGHT anywhere in the canvas, also. Note that some items in the menu are not available (they are "grayed out"); as you select devices in the map, other items in the menu will be available. When a server is selected, for example, Information, Free Devices, and the Note Pad are always available. Each of the View menu items are described below.

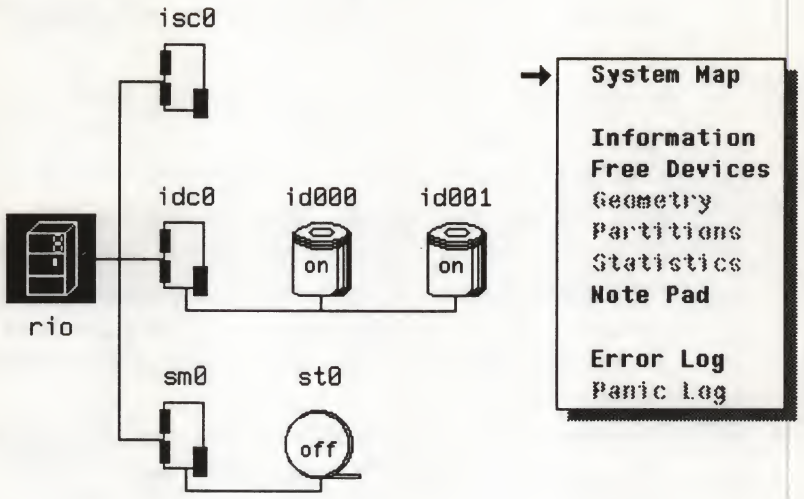


Figure 6-4 *View Menu: Server Selected*

Popup Windows

Once you select a device, you can select an item in the View menu to display information about the device; when you select a menu item, a window will *pop up* — normally at the top right of the primary window.

There are three types of windows: informational (General Information, Free Devices, Geometry, Partitions, and Statistics), Note Pad, and Error Log (which includes panic and all other types of errors).

Informational Popups

Information: Information provides different information for each type of device selected. For a server, there is much information available: Ethernet address, host ID number, hardware type, screen size, and much more.* For a tape drive, there is a little information: tape type and error information, mostly, and the information is available only if a tape is mounted. For a tape controller, there is no information.

Free Devices: Available only when the server is selected, Free Devices list the attached *free* devices (meaning they have no controller) on the system. Examples are Ethernet, graphics boards, PROMs, color or monochrome monitor connections, tty drivers, and floating point controllers.

Geometry: Available only when a disk drive is selected, Geometry lists the raw capacity of a disk drive; the numbers of physical and data cylinders, alternate cylinders, data sectors, and heads; plus the cylinder offset, interleave factor, and revolutions per minute.

* Note that much of the information comes from the EEPROM, and may not be accurate if you have not programmed your EEPROM. The "Last hardware update," for example, has a default value of December 31, 1969, the beginning of the Unix epoch. See the *PROM User's Manual* and the `eeprom(8s)` man page for instructions for updating the EEPROM.

Partitions: Available only when a disk drive is selected, Partitions lists the partitions on a disk, telling you the number of blocks in the partition, its size in megabytes, the starting cylinder number of the partition, and the number of cylinders in the partition.

Statistics: Available only when a disk drive is selected, Statistics shows you the number of reads, writes, and I/Os since the last reboot, plus the total number of kilobytes transferred and the average milliseconds per I/O.

Note: with SunOS 4.0.3 you may only view statistics for the first four disks on the server. With SunOS 4.1 statistics for up to ten disks can be viewed. Note further that you may need to reconfigure your kernel, eliminating non-existent devices, to allow you to see statistics for even four disks. See "Eliminating Non-Existent Devices" near the end of Chapter 2 for detailed instructions.

Note Pad Popup

Note Pad: All devices have a Note Pad; use it as a place for engineering notes — to type in pertinent information about the device, including such things as installation date, location, applications, habitual problems, service record, etc.

There is no required format for a Note Pad, but the first entry often says what type of device this is. The Note Pad is a standard SunView text subwindow, so you can manipulate text there just as you can in other text subwindows.

Note that the cursor is at the top of the Note Pad when you open it; use the scrollbars to scroll to another location and click LEFT to reposition the cursor if you want to enter text elsewhere.

Error Log Popups

Error Log: Available for any device that has any errors logged, the Error Log encompasses all types of errors logged on the system (panic, severe, critical, non-critical, warning, notice, and informational).

Note that IOadmin `syslogd` takes all system-generated messages and separates them into the seven categories listed above, places the message in the logfile for its category — *and* into the log files for all errors whose severity is lower than it. For example, the most severe category of error — Panic* — is logged in the Panic Log and in all other logs, since their severity is lower than it. The Informational Log contains every message that `syslogd` logs.

Panic Log: Only panic errors are included in the Panic Log. Panic errors often indicate a serious problem that needs immediate attention, so Device Management treats them in a special way. If any panic errors are logged while you are running Device Management, or if any panic errors have been logged since the last time you ran Device Management, the device which logged the error will be flagged with a bold exclamation point (!) on top of its icon.

* Note that the Panic Log does not log system panics, which usually imply a system crash. The Panic Log includes emergency error messages for a device, which often precede device failure.

Displaying an Error Log

If you have started the `dmt syslogd`, as described in "The Do Menu" below, device errors will be logged and saved in the seven `dmt` categories. When you select Error Log from the View menu, there is a delay while the error logs are parsed, and then a popup comes up, with the Informational Log displayed.

To view the errors in a different category, press RIGHT on the View button and select the error type you wish to view.

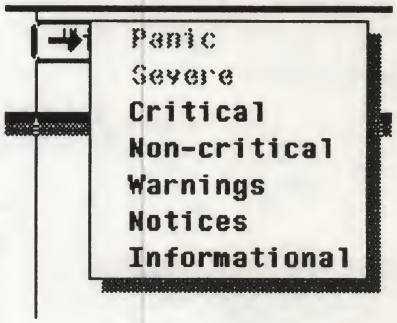


Figure 6-5 *View Errors Menu*

Clearing Panic Error Flag

To ensure that panic errors receive your attention, the panic error flag (!) remains on top of the icon of the device that logged the error until you click on the Reset Error Flags button, which is displayed in the Panic Log window. Note that clicking on the Reset Error Flags button clears the exclamation point from all devices that were so marked.

To view the log, select the device by clicking LEFT on its icon, press RIGHT in the canvas to display the View menu, and select Panic Log. The error flag will be reset when you click on the Reset Error Flags button.

Note that unless you specify a different interval with the `-t` option when you start Device Management, the panic log is checked every five minutes; panic errors may not show up immediately. Check the server's console messages if you think there may be a problem; if the system slows down significantly, it may be logging panic errors.

6.5. The Do Menu

Press RIGHT on the Do button to display the Do menu, which has four options: Update Display, Logging Status, Disk Test, and File Test. Note that Disk Test and File Test are not executable unless a disk is selected on the system map. Note also that you have to be the superuser to execute Disk Test and File Test.

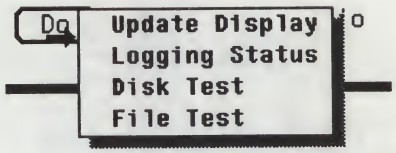


Figure 6-6 *Do Menu*

Updating the Display

Press RIGHT on the Do menu and select Update Display to force a check on the status of displayed devices. This is the same function which is performed every five minutes by default. When you select Update Display there will be a delay while information is gathered and the Device Management window will be redrawn, indicating any status changes that might have occurred since the last update.

Starting dmt Logging

Press RIGHT on the Do menu and select Logging Status to display the Logging Status popup window. If `syslogd` is running on the server, as it likely will be, the Process Name field will indicate that `syslogd` is running, a Process ID will be displayed, and a date will be shown in the Logging Since field. Unless you have selected Start in the Logging Status window at some time on the current server, the `syslogd` running will be the standard `syslogd` and not the special `dmt syslogd`.

Click LEFT on Start to start the `dmt syslogd` running. This will kill the currently-running `syslogd`, if any, and start a new `syslogd`, using the IOadmin configuration file (`syslog.conf`), thus enabling IOadmin error logging. The new status will appear in the status fields: a new Process ID will be shown, and the Logging Since time will have changed.

If you wish to stop the IOadmin `syslogd` and restart the standard `syslogd`, as root kill the process from the command line and start the system `syslogd` by typing `/usr/etc/syslogd&`.

Note that you can tell if the `dmt syslogd` is running by looking at the result of a `ps` command; if the `dmt syslogd` is running, the `-f` flag will be displayed:

```
% ps aux | grep syslogd
  . . .
/usr/etc/syslogd -f/var/spool/ioadm/dmtlog/syslog.conf
```

Modify the IOadmin `syslogd` startup line in `/etc/rc.local` if you want to enable `dmt` logging automatically at bootup, as shown in Chapter 2.

Running Disk Test

The Disk Test checks to see that a disk is accessible through the use of read(2V) and write(2V) system calls. The tests are generally non-destructive, *but must be run on unmounted file systems to ensure that no harm is done*. See "File System Management" (Chapter 7) for information about mounting and unmounting file systems.

1. **Select a disk by clicking LEFT on it.**
2. **Select Disk Test in Do menu.**

   The Disk Test popup menu will be displayed.
3. **Choose test options.**

   Options to be chosen include partition to be tested, type of read or write test, test pattern, number of analysis passes, whether to stop or continue if an error is detected, and whether you want verbose or concise error reporting. Select cycle-button choices by clicking LEFT on the cycle button until your choice is displayed, or by pressing RIGHT, selecting your choice, and releasing the mouse button.
4. **Click Run to start the test.** If a password is required to become superuser on the server, a popup will be displayed, asking you to enter the server's superuser password.

   The test will start, with the command and test information displayed in the subwindow below the function buttons. As it runs, you will see progress reports: "Test completion 1%, 2% . . ." etc.

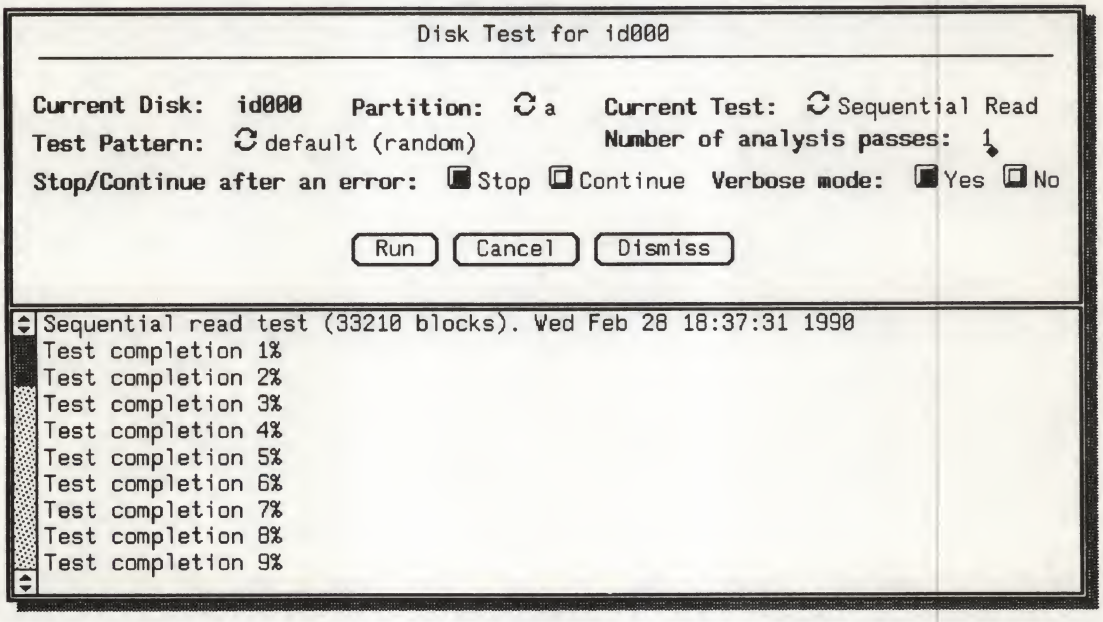


Figure 6-7 *Disk Test Popup Window*

You can cancel at any time by clicking LEFT on the CANCEL button.

When the test is complete, a completion message will be displayed. The results of the test are saved in /var/spool/ioadm/dmtlog.

Running File Test

File Test checks to see that a file system is accessible through the use of open(2V), close(2), read(2V), and write(2V) system calls. *The file system must be mounted during this test.*

1. **Select File Test in Do menu.**

   The File Test popup window will be displayed.

2. **Choose test options.**

   Options to be chosen include partition to be tested, test pattern, number of analysis passes, size of file (*filetest* creates two files in file system, writing in them with test pattern designated; test continues until files are this size), and whether you want verbose or concise error reporting. Select cycle-button choices by clicking LEFT in the area of the cycle button until your choice is displayed, or by pressing RIGHT, selecting your choice, and releasing the mouse button.

3. **Click Run to start the test.** If a password is required to become superuser on the server, a popup will be displayed, asking you to enter the server's superuser password.

   The test will start, with the command and test information displayed in the subwindow below the function buttons. As it runs, you will see progress reports: "Test completion 1%, 2% . . ." etc.

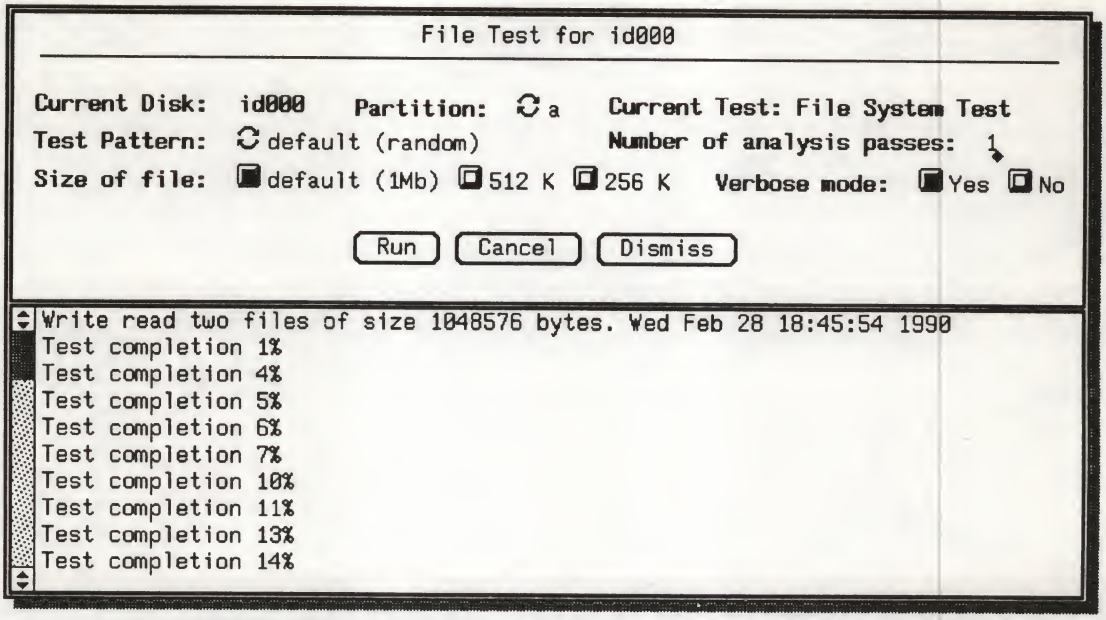


Figure 6-8 *File Test*

You can cancel at any time by clicking LEFT on the CANCEL button. When the test is complete, a completion message will be displayed. The results of the test are saved in /var/spool/ioadm/dmtlog.

7

File System Management

File System Management helps you manage files and UFS file systems:

- graphically displays information about disk partitions and file systems
- displays detailed information about files and directories
- invokes `newfs` for creating new file systems
- invokes `fsck` to check file systems
- invokes `mount` and `umount` to mount or unmount file systems.

In addition, File System Management allows you to view, activate, clear, and resync MetaDisk metapartitions. See Chapters 3 and 4 for detailed information about the MetaDisk.

7.1. Terminology

- **disk**: a physical disk, such as an IPI (`id000, id001`, etc.) or a Xylogics (`xy0, xy1`, etc.)
- **formatting a disk:** the act of defining *partitions* on a disk and preparing the disk for reading and writing.
- **disk partition:** composed of a variable number of contiguous cylinders of a disk, a partition is defined by the `format` command. There can be as many as eight partitions on a disk, named a to h (xy0a, xy0b, . . . xy0h). A UNIX file system can be no larger than a single partition. File systems are created and mounted on partitions. Traditionally, partition `a` on the first disk is used as the root partition, partition `b` as the swap partition, and partition `c` comprises the entire disk.
- **file system:** area on a disk where files can be written and read.
- **creating a file system:** the act of defining a file system on a disk partition with the `newfs` command.
- **file system check (`fsck`) :** a program — `fsck` — for checking the integrity of a file system.
- **mounting a file system:** making a file system available for reading and writing by *mounting* it on some directory in the UNIX directory hierarchy; this *mount point* is usually defined in the file `/etc/fstab` and read during the boot process.

- **/etc/fstab:** the file which contains entries for file systems and disk partitions to mount using the `mount` command, which is normally invoked by boot scripts at boot time.
- **mtab:** a table of file systems currently mounted.
- **unmounting a file system:** making a mounted file system unavailable for reading and writing.
- **mount point:** the directory in the UNIX hierarchy where a file system is mounted.

7.2. Starting File System Management

Start File System Management as described in Chapter 5: type `fsm` and press Return:

```
% fsm&
```

After a few moments, the primary File System Management window will be displayed in the center of your screen. If you have entered server names in `.ioadminhosts` in your login directory, the name of the first server in the list will be displayed in the Server Name field; otherwise your machine name will be displayed. If you start `fsm` with the `-h` flag, followed by a number of host-names, the first name following the `-h` will be in the Server Name field. Press RIGHT in the area of the Server Name field to see other available servers.

You can start the IOadmin tools with special SunView characteristics, as described in the *SunView User's Guide*; you can designate the color of the windows, for instance, or the font to be used for text.

In addition, there are three IOadmin options which can be added to the command-line startup (include a space between the option and its argument, if any):

-h [*servername1 servername2 servername3 etc.]*: start the tool with the named servers preselected when the tool comes up; each server will appear in the Server Name pulldown menu.

-m: run the tool in monochrome on a color monitor.

-t *seconds*: change the default timing interval from 300 seconds to another value. This is the interval for updating dynamic displays for Device Management and File System Management.

If you are not in the IOadmin tools directory or if you have not set your path variable to include the IOadmin tools directory, as described in Chapter 5, include the full path name.

7.3. Closing or Quitting File System Management

When you are finished using `fsm`, you can close the tool to an icon or quit the tool altogether.

Closing File System Management to an Icon

Press RIGHT anywhere in the frame border and select Close; File System Management will be closed to an icon on the desktop, with the name of the current server displayed.

Figure 7-1 *File System Management Icon*

Quitting File System Management

Press RIGHT anywhere in the frame border and select Quit. You will be asked if you really want to quit; click LEFT on the Confirm button to quit, or LEFT on the Cancel button if you don't want to quit. When you click LEFT on Confirm, the primary tool window and all subwindows will disappear.

7.4. The View Menu

Press RIGHT on the View button (or press RIGHT anywhere in the canvas) to display the View menu; move the mouse to the item you wish to view and release the mouse button to select it.

There are three items in the View menu related to disk partitions: Disk Layout, Partitions (which includes more entries in a pull-right menu), and Selection Detail. Note that Selection Detail is not available until you select a partition (explained below).

Disk Layout

The Disk Layout, which is the starting point for most File System Management activities, is a graphic representation of all of the disks and defined disk partitions on a system. It shows all of the disks, one per row, in alphabetic order, with all of their partitions delineated for selection.

Metapartition names start with "md.". On a color monitor, metapartitions and their physical components are pink; other partitions are blue. On a monochrome monitor, metapartitions and their physical components are striped; if the metapartition is inactive (not activated with `metainit`), the striping will be "broken," as shown in `md0` below.

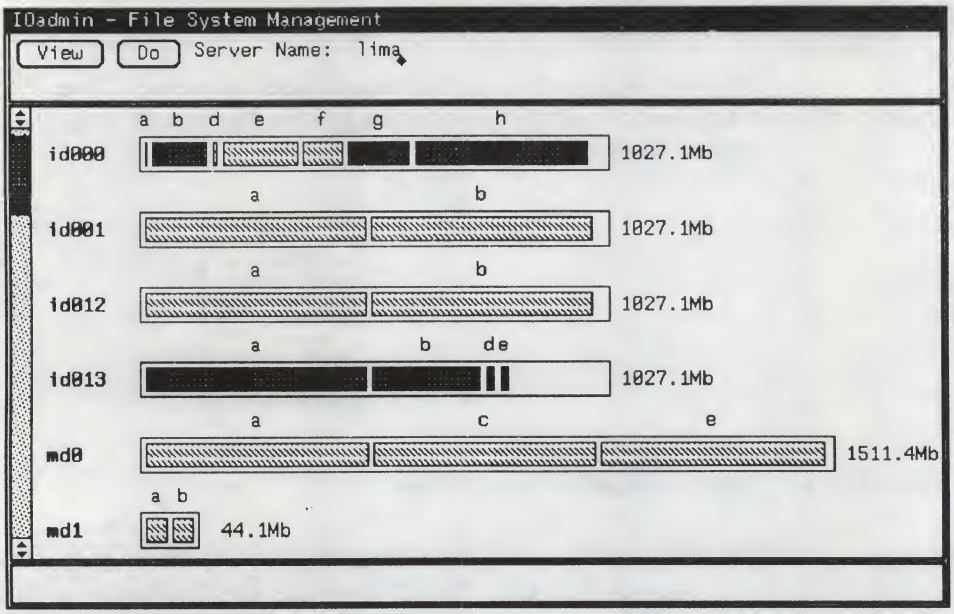


Figure 7-2 *FSM: Disk Layout*

Disk Layout — Metapartition Selected

In the following figure, a metapartition is selected; note that the physical partitions (`id001b` and `id001h`) that comprise the metapartition (`md1a`) are shaded.

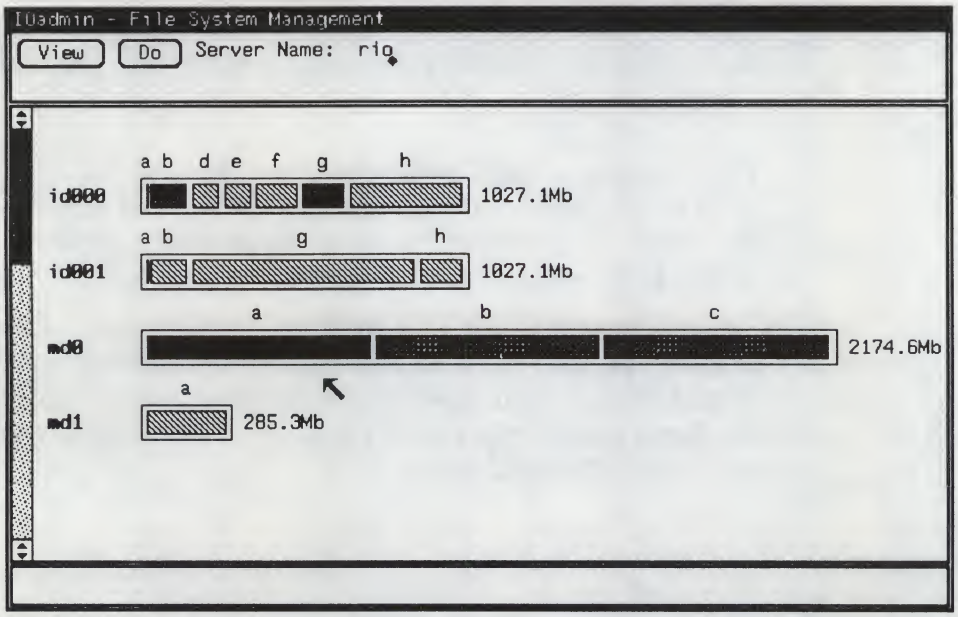


Figure 7-3 *FSM: Disk Layout - Metapartition Selected*

Partitions

This is an expansion of Disk Layout, with each partition shown beneath the previous partition, and with the size and use of the partition shown to the right of the partition. There are five categories of partition display.

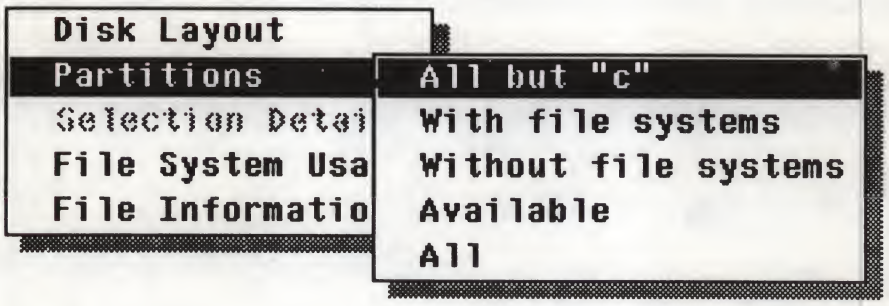


Figure 7-4 *FSM: Partition Menu - All but "c" Selected*

Partitions - All but "c"

Select this item to show all defined partitions, including metapartitions — but not including the c partitions of physical disks.

Partitions - with File Systems

This category includes all partitions on which a file system has been detected. This detection is done by reading the partition's superblock and looking for a special number (file system magic number). Note that partitions and metapartitions with file systems defined are shown as "not mounted" or the mount point is shown.

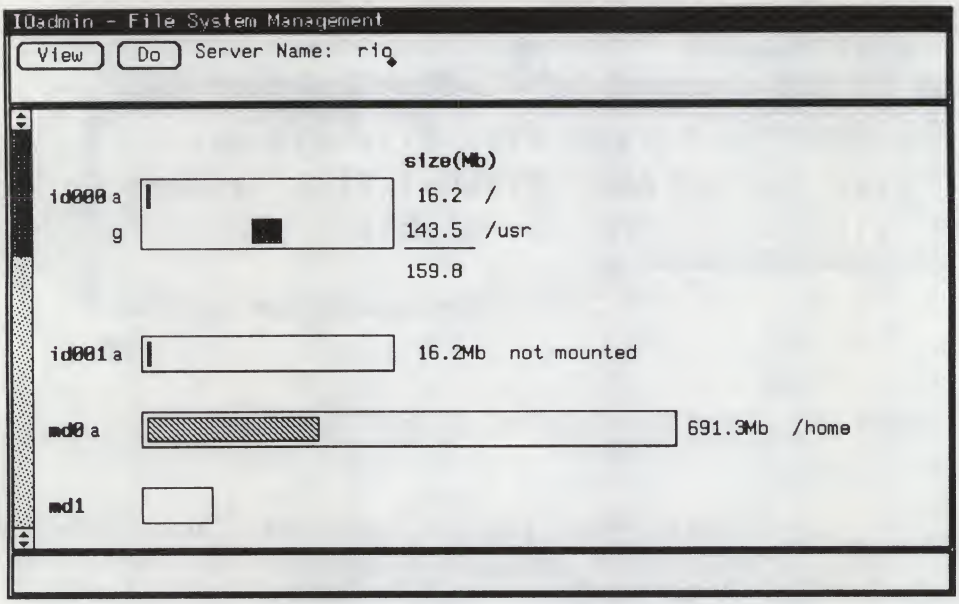


Figure 7-5 *FSM: Partitions With File Systems*

Partitions - without File Systems

This category includes all partitions that are not defined in `/etc/fstab` nor shown in `mtab` as mounted on a file system; could be swap or available.

Partitions - Available

Available partitions are those that are not designated for swapping nor defined as a file system in `/etc/fstab` nor shown as mounted in `mtab`. Only one partition — `xy0e` — is available in this example.

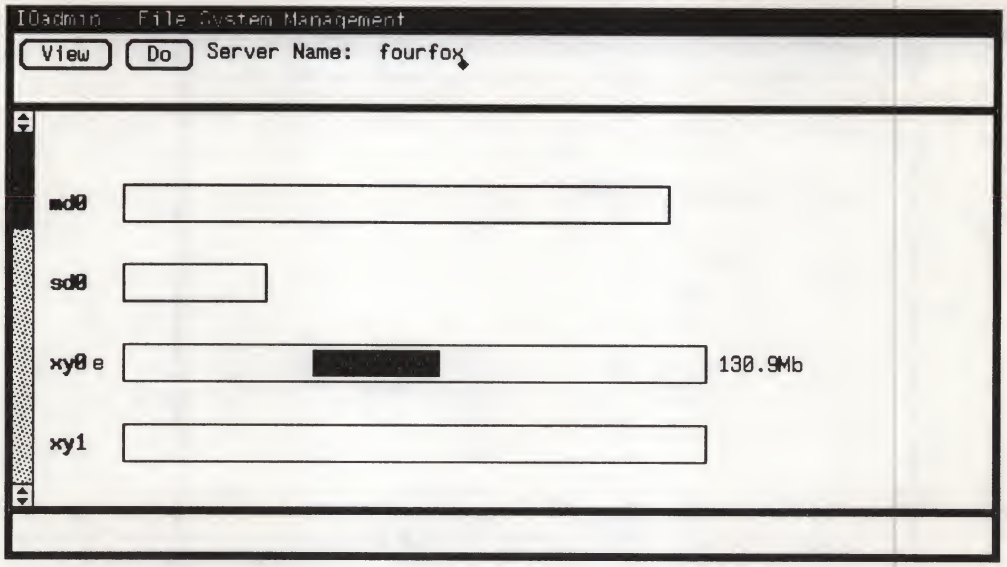


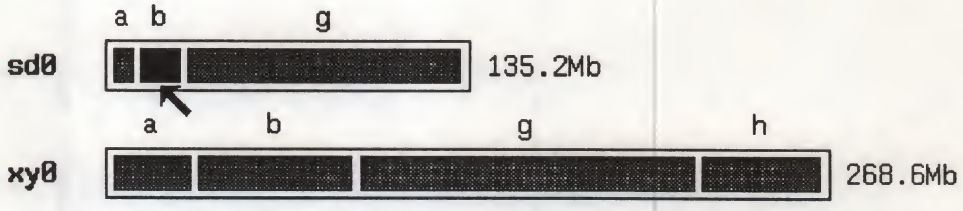
Figure 7-6 *Partitions - Available*

Partitions - All

Select this item to show all defined partitions, including metapartitions. If any two partitions overlapped, which is possible, you would immediately see that in this chart.

7.5. Selecting a Partition

To see detail about a partition, click LEFT on the partition to select it (from the Disk Layout or from one of the Partitions displays). Once you have selected a partition, Selection Detail will be available in the View menu.



Selecting Multiple Partitions

To select a number of partitions at one time, click LEFT on one partition to select it from the Disk Layout; then click MIDDLE to select additional partitions. Once you have selected a number of partitions, Selection Detail will be available in the View menu; you can view detail for all selected partitions at one time.

7.6. Selection Detail

Select Selection Detail to display information about the selected partition in a popup window.

Physical Partition

In the figure below, a physical disk partition is shown.

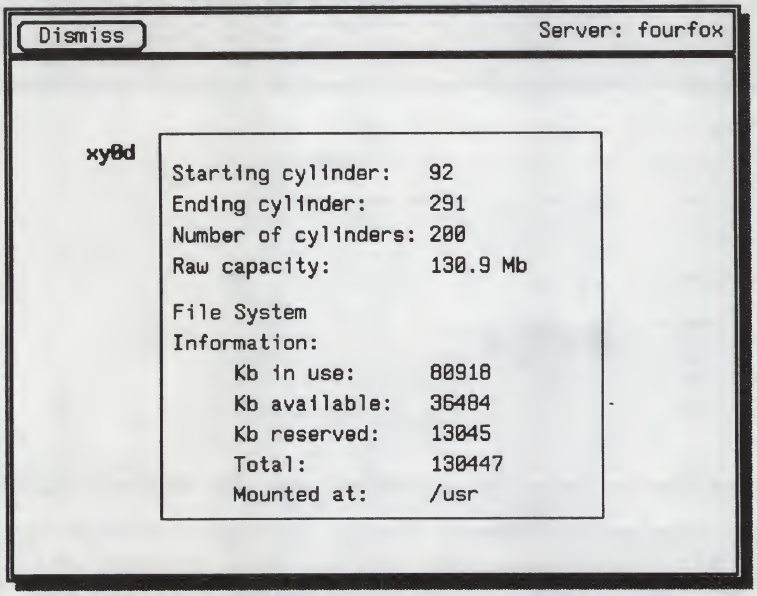


Figure 7-7 *FSM: Selection Detail - xy0d*

Metapartition Detail

In this example, a metapartition is shown. Note the pie chart, showing the physical partitions that comprise the metapartition.

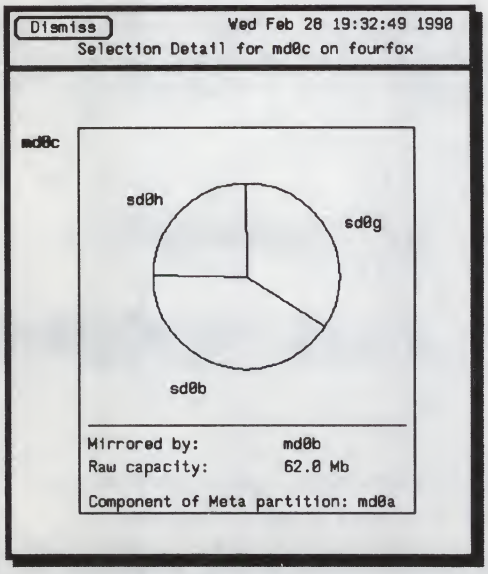


Figure 7-8 *Selection Detail - md0c*

7.7. File System Usage

File System Usage shows you information similar to the `df -t4.2` command, and includes space reserved for superuser, in graphic and text form. MetaDisk information is also provided. The dotted portion of the bar represents the reserve capacity of the file system — that part of the file system available when it is 100% used.

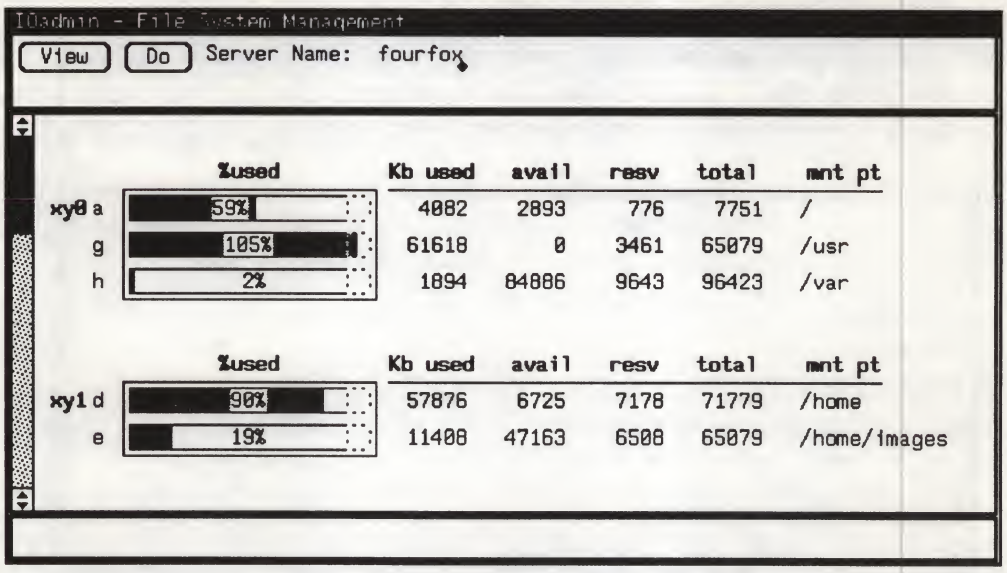


Figure 7-9 *File System Usage*

7.8. File Information

File Information tells you all you could want to know about a file or directory. Note that gathering the information can take a significant amount of time if you select a large directory such as `/usr`. The information displayed is similar to that displayed by the `du` and `ls` commands (with all `ls` flags included).

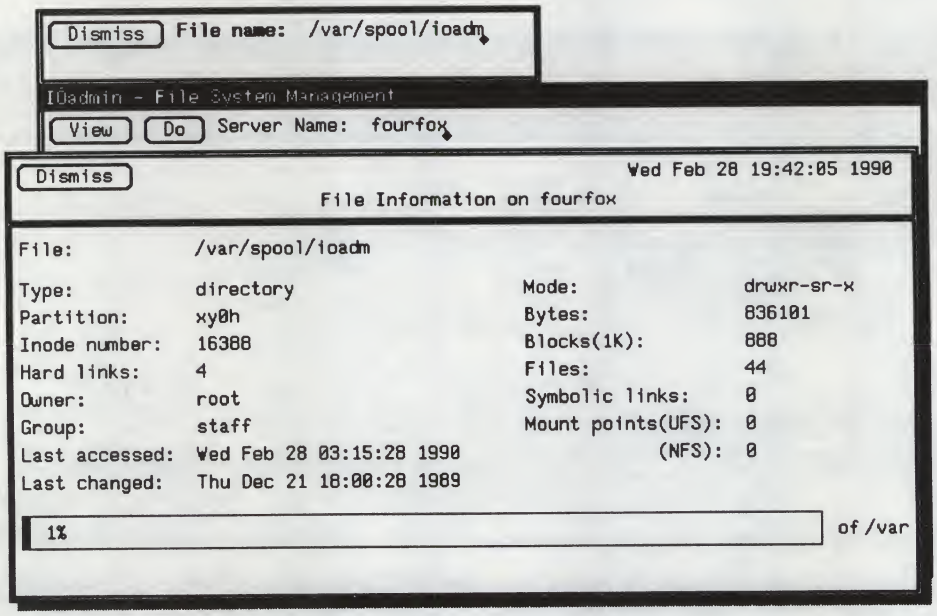


Figure 7-10 *File Information*

7.9. The Do Menu

The only available item in the Do menu is "Update display" until you select a partition.

NOTE: you must be `root` to perform certain functions (Create File System, Check File System, Mount File System, and all three metapartition functions). Be sure you know what you are doing when you select one of these functions; these functions — especially Create File System and the metapartition functions — can have serious consequences.

Note also that you continue to have superuser privileges on the current server until you select another server or quit `fsm`.

Update Display

By default there is a five-minute interval between automatic updates of the File System Management display. Select Update display in the Do menu to update the display after you have made changes to any partition. If you create a new file system, for instance, or create a metapartition, the changes will be reflected only after you select Update display (or press Return with the pointer in the header panel).

You can start `fsm` with a different update period by including the `-t` argument, as described in Chapter 5.

Create File System

Do the following to create a new file system on an available disk partition.

1. **Select an available partition.**

   An *available* partition is one without a file system, one not used as swap, or one that is a component of a metapartition.

2. **Select Create File System in Do menu.**

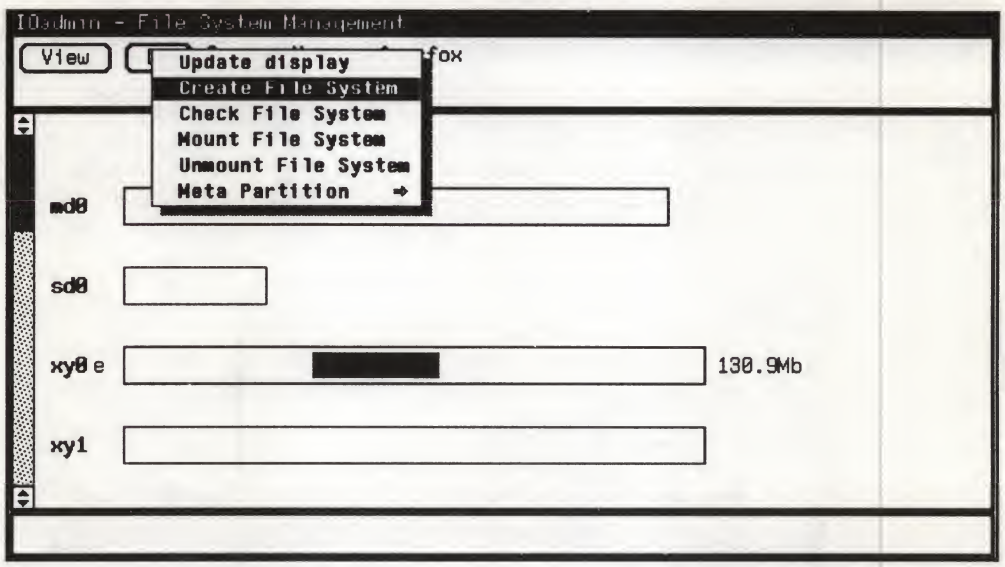


Figure 7-11 *Create File System Selected*

An alert message box will pop up, telling you that you are about to create a new file system on a specified partition of the specified server; you can click LEFT on Confirm to continue or click LEFT on Cancel to cancel the procedure.

Note that you cannot create a file system on a designated swap partition or on a metapartition component.

3. **Click LEFT on Confirm to continue.**

   A window will pop up, displaying the UNIX command that is to be run (in this case, a new file system will be created on device sd0b on the server four-fox):

Check File System

To perform a file system check on a file system partition, do the following:

1. **Click LEFT on file system partition to be checked.**
2. **Click LEFT on Check File System in Do menu.**

   If you did not start `fsm` as root or if you have not entered the root password for this server, a popup will be displayed, asking you to enter the root password. After you enter the root password, one of two windows will pop up: if the selected partition is mounted on a file system, a warning popup will be displayed:

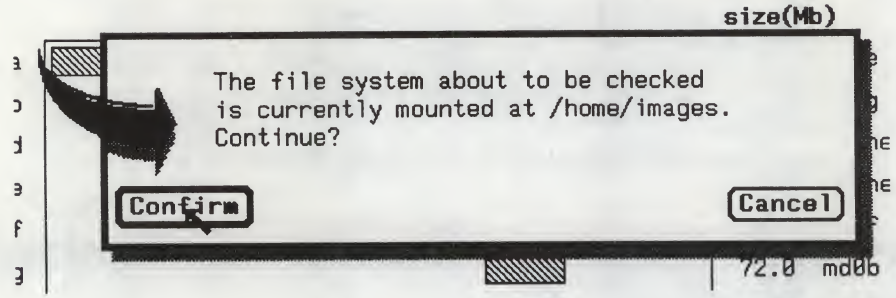


Figure 7-12 *File System Check Warning*

After you click "Confirm" — or if there is no file system mounted, another window will pop up, displaying the results of the test.

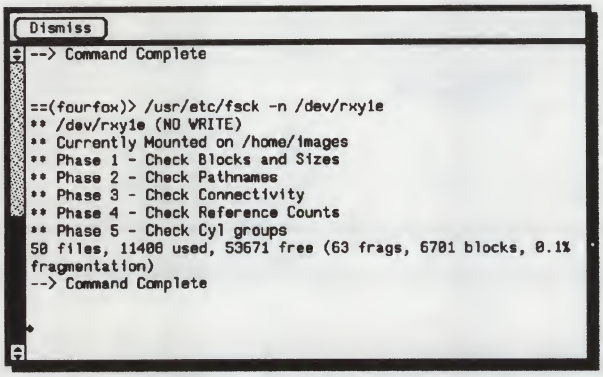


Figure 7-13 *File System Check*

NOTE: `fsck` will be run in `-n` mode; basically, this means that any problems encountered will be listed, but will not be fixed. See the man page for `fsck` for details.

Mount File System

To mount a file system:

1. **Select file system partition in Disk Layout or Partition display.**
2. **Select Mount File System in Do menu.**

   A subwindow will pop up, waiting for you to type in the full path of the directory where the file system is to be mounted; move the pointer to the mount window before typing. If you did not start the tool as root or if you have not entered the root password for this server, you will be asked for the root password before the mount window pops up.

Note: if the file system is already mounted, if the selected partition is designated as swap, or if the partition is a component of a metapartition, the request will be ignored and a message to that effect will be displayed in the message panel of the File System Management window.

3. **Type in directory path and press Return.**

   The mount command will be issued (in this case, mount xy0f at `/mnt` on server fourfox):

Unmount File System

To unmount a file system:

1. **Select file system partition in Disk Layout or Partition display.**
2. **Select Unmount File System in Do menu.**

   If you did not start the tool as root or if you have not entered the root password for this server, you will be asked for the root password before the mount window pops up. A subwindow will pop up, displaying the `umount` command that was executed.

   Note: if the selected partition is in use, the file system will not be unmounted.

Metapartitions

Metapartitions created with the MetaDisk Driver can be displayed just as can real partitions, and file systems can be created, checked, mounted, and unmounted on metapartitions just as on real partitions.

Note that metapartitions and their component physical partitions have a distinctive pattern or color; select a metapartition to see which physical partitions comprise it. The physical partitions will be shaded.

Activating, Clearing, or Resyncing Metapartitions

The Meta Partition item in the Do menu has three MetaDisk functions beneath it: Activate, Clear, and metasync. To perform one of these function, do the following.

1. Select a metapartition.
2. Press RIGHT on Meta Partition, move the mouse to select the function you want, and release the mouse button. In this example, `metasync` was selected; note that `metasync` is selectable only when a metamirror component has been selected, and that the metapartition selected is considered to be the good metapartition. The `metasync` is done *from* the selected metapartition to the other metapartition.

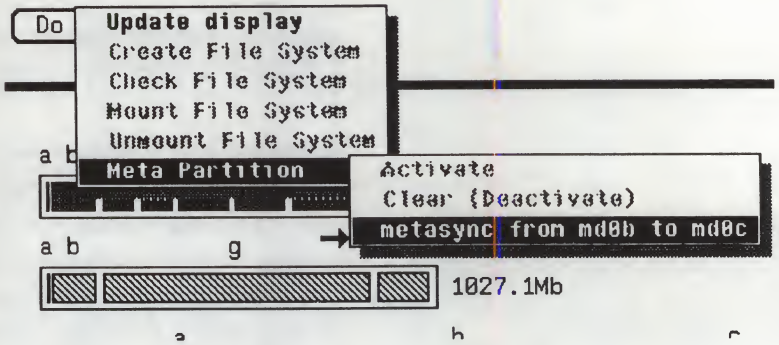


Figure 7-14 *Do Menu: Metapartition Resync*

If you did not start the tool as root or if you have not entered the root password for this server, you will be asked for the root password before the window displaying the function performed pops up. The selected function will be performed. See Chapters 3 and 4 for more about the MetaDisk Driver and about the three functions.

8

Performance Monitor

Performance Monitor is a monitoring tool; it uses the same information as *PerfMeter*, but presents the data in more detail, and it displays both current and historical data. Use Performance Monitor to see the following types of information presented with bar graphs, line graphs, and tables:

- CPU Usage
- Disk Transfers
- Paging Activity
- Swapping Activity
- Number of Device Interrupts
- Number of Packets
- Number of Packet Errors
- Number of Packet Collisions
- System Load Average

8.1. Starting Performance Monitor

Start Performance Monitor from the command line as described in Chapter 5: type `pmt` and press Return:

```
% pmt&
```

If you have entered server names in `.ioadminhosts` in your login directory, the name of the first server in the list will be displayed in the Server Name field; otherwise your machine name will be displayed.

You can start the IOadmin tools with special SunView characteristics, as described in the *SunView User's Guide*; you can designate the color of the windows, for instance, or the font to be used for text.

In addition, there are three IOadmin options which can be added to the command-line startup (include a space between the option and its argument, if any):

-h [*servername1 servername2 servername3 etc.]*: start the tool with the named servers preselected when the tool comes up; each server will appear in the Server Name pulldown menu.

-m: run the tool in monochrome on a color monitor.

-t *seconds*: change the default timing interval from 300 seconds to another value. This is the interval for updating dynamic displays for Device Management and File System Management.

Press Return with the pointer in the header panel to view current data on the machine named, or press RIGHT in the Server Name field to select another server. See Chapter 5 for more information about IOadmin tools.

If you are not in the IOadmin tools directory or if you have not set your path variable to include IOadmin tools, include the full path name to `pmt` (`/usr/ioadm/bin/pmt` by default).

8.2. Closing or Quitting Performance Monitor

Closing Performance Monitor to an Icon

Press RIGHT anywhere in the frame border and select Close; Performance Monitor will be closed to an icon.

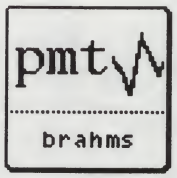


Figure 8-1 *Performance Monitor Icon*

Quitting Performance Monitor

Press RIGHT anywhere in the frame border and select Quit. You will be asked if you really want to quit; click LEFT on the Confirm button to quit, or LEFT on the Cancel button if you don't want to quit. When you click LEFT on Confirm, the primary tool window and all subwindows will disappear.

8.3. The View Menu

The View menu is the primary focus of Performance Monitor; it is here that you designate whether you see logged or current data, the format in which you will

see the data, and which of the nine categories of data will be displayed.

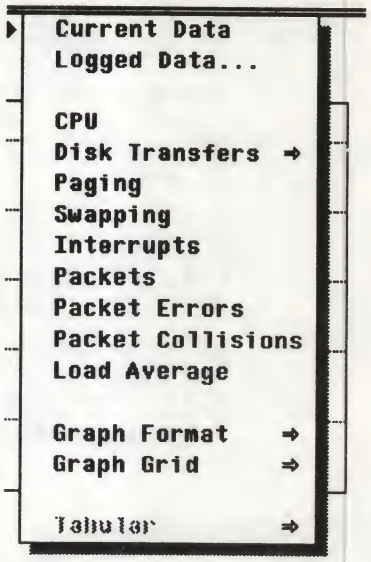


Figure 8-2 *View Menu*

Logged or Current Data?

Select Logged Data to view historical data — data that has been logged previously, as described in "Logging" below. Select Current Data to view current, dynamic data in any of the nine categories. The default is current data.

Bar or Line Graph?

Press RIGHT on Graph Format and move the mouse to the right to select bar or line graph format. You can also click LEFT anywhere on the graph itself to toggle back and forth between bar and line format. The default is line format.

Click on a disk when viewing Disk Transfers in bar format to see the line graph for the selected disk.

Grid On or Off?

Press RIGHT on Graph Grid and move the mouse to the right to select grid On or Off. The default is On.

Tabular Data?

For logged data you can view the values that created the chart — in either graph data or raw data format. Press RIGHT on Tabular and move the mouse to the right to select either Graph Data or Raw Data. A window will pop up, displaying the data requested in a text subwindow.

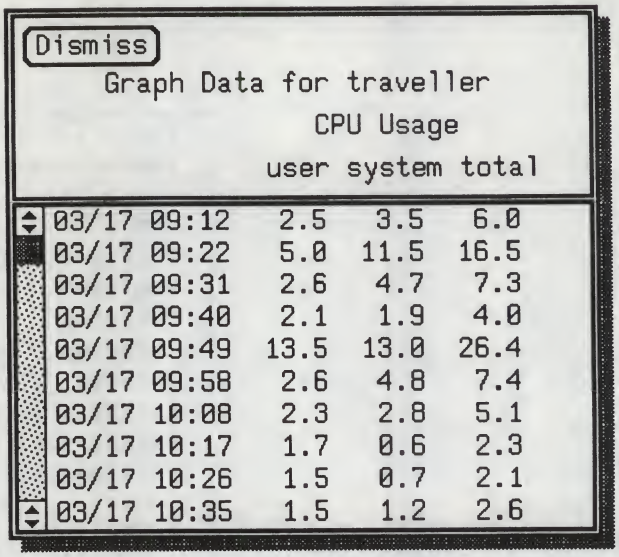


Figure 8-3 *Graph Data*

8.4. Viewing Current Data

For problem solving you will normally use view logged data, but for a quick look at what is happening now, select Current Data. The screen is updated every two seconds, so it doesn't take long to see trends. You might want to display CPU Usage, for instance, and then run a CPU hog such as `troff` in another shell to see what happens to the chart.

The graphs are self-explanatory. Note that the labels on the x and y coordinates change, depending on what is being displayed. In the Paging chart shown below, for example, the number of I/O pages per second (from 0 to 55) is shown.

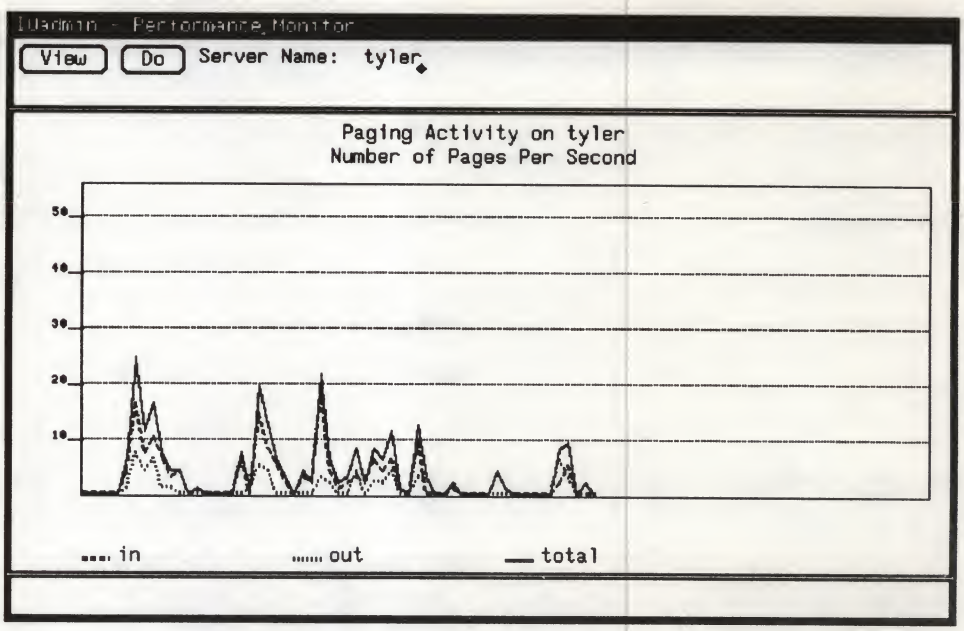


Figure 8-4 *Paging Activity*

8.5. Logging

Before you can view logged data, you must have created some logged data. Once you have been logging for a while, you can view data from any day, and any period during any day.

Start Logging

1. **Select Logging Status in Do menu.**

   The Logging Status window will pop up.

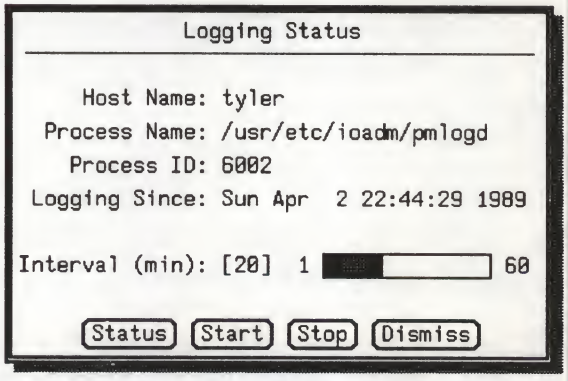


Figure 8-5 *Logging Status Window*

2. **Click LEFT on Start.**

3. Enter the root password when the root password popup is displayed.

   After a short delay, logging should start; the name of the current host (server), the logging process name (`/usr/ioadm/etc/pmlogd`), and the process ID number will be displayed. Unless you stop logging explicitly or there is a system problem or crash, you do not need to start logging again.

Stop Logging

If you wish to stop logging for any reason, do the following:

1. **Select Logging Status in Do menu.**
2. **Click LEFT on Stop.**

   Enter root password, if requested.

Change Logging Interval

The default logging interval is 20 minutes (an entry is made in the log file every 20 minutes). If you wish to change that interval, do the following:

1. **Select Logging Status in Do menu.**
2. **Click or press LEFT on the Interval icon to increase or decrease the interval.**
3. **Click LEFT on Stop.**

   Enter root password, if requested.
4. **Click LEFT on Start.**

   Logging will begin again, using the new interval.

8.6. Disk Storage Considerations

Performance Monitor logs, which are stored in `/var/spool/ioadm/pmlog`, are between 8000 to 10000 bytes long per day if you use the default sampling period of 20 minutes; the shorter the period, the longer the logs. If you have disk storage problems, you may want to delete logs, or copy them to tape or some other location on disk.

8.7. Viewing Logged Data

You can view logged data for a specified time period or you can load a specific log file and view the data there.

For a Time Period

1. **Select Logged Data in View menu.**

   The Logged Data window will pop up:

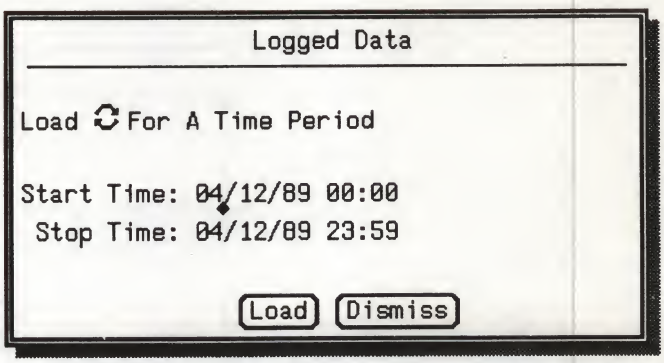


Figure 8-6 *Logged Data Window*

2. **Select start and stop time.**

   Click LEFT on month, day, year, hour, or minute to change any of these fields; the field will be highlighted. Delete the current value and type in the new value for each field separately.

3. **Click LEFT on Load.**

4. **Select What to View.**

   Select from one of the nine categories of information and release the mouse button.

Load a Specific File

Under some circumstances you may want to load a specific file instead of viewing data for a time period. If you have copied a file from another server, for instance, you can view the data in this way.

1. **Select Logged Data in View menu.**

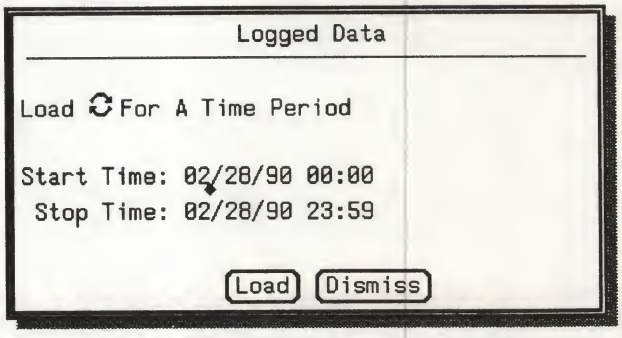


Figure 8-7 *"Load For a Time Period" Selected*

2. **Click LEFT on cycle button.**

   The header text will change from "Load For a Time Period" to "Load a

Specific File," with a place for you to type in a directory and file name.

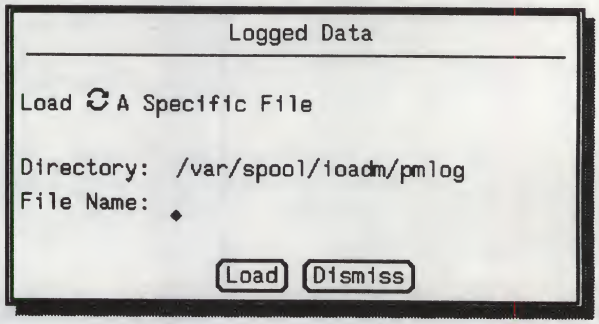


Figure 8-8 *Load A Specific File*

3. **Type in name of file.**

   Assuming the file you wish to load is in `/var/spool/ioadm/pmlog`, the default directory, type in the file name only. If the file you wish to load is in another directory, click LEFT on the directory name, delete it, and type in the correct directory.

4. **Click LEFT on Load.**

5. **Select What to View.**

   Select from one of the nine categories of information and release the mouse button.

CPU Utilization - Bar Graph

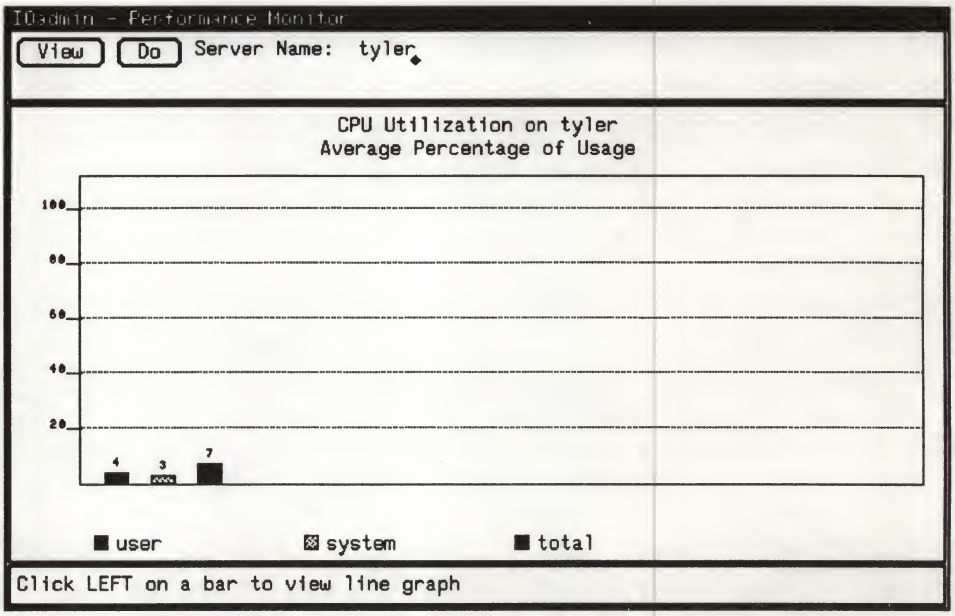


Figure 8-9 *CPU Utilization: Bar Graph*

CPU Utilization - Line Graph

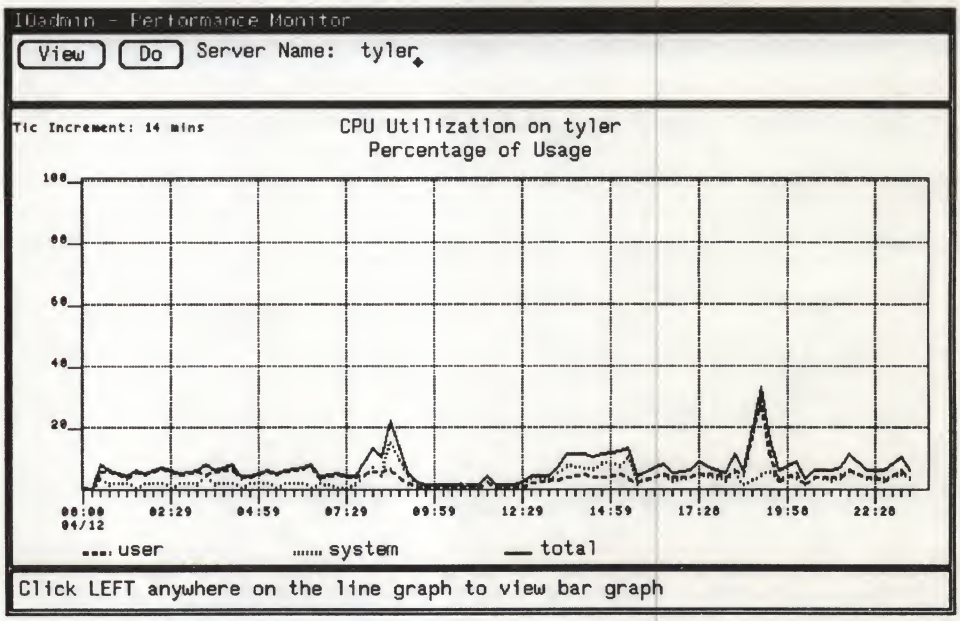


Figure 8-10 *CPU Utilization: Line Graph*

APPENDIX A: Man Pages

NAME

dmt – IOadmin Device Management (SPARCserver Manager)

SYNOPSIS

dmt [**–m**] [**–h** *hostname* [*hostname*]] [**–t** *seconds*]

AVAILABILITY

This tool is only available with the IOadmin tools which are part of the SPARCserver Manager software package.

DESCRIPTION

dmt is a SunView tool which, for a specified host, allows the user to:

- View system configuration of controllers, disk drives, and tape drives
- View information about servers and controllers, plus geometry, partitioning, and statistics about disks
- View error and panic logs written to by **syslogd**(8)
- Status **syslogd**(8)
- Check accessibility of disks and file systems by running DiskTest and FileTest

The data available through **dmt** is similar to that available through **dkinfo**, **hostid**, **eeprom**, **domainname**, and **getservbyname**, but is displayed in greater detail.

OPTIONS

–m Run in monochrome. Default is to run in color on color systems, monochrome on monochrome systems.

–h *hostname*

Come up ready to access *hostname*. The default is the local system. More than one *hostname* may be specified. A menu of names will be created from which to choose a server. Once the tool is started the user may change the host name at will.

–t *seconds*

Update the screen display every *seconds*. The default is 300 (5 minutes). To disable periodic update, use a value of zero (e.g. dmt -t 0).

USAGE

For **dmt** to access a host, the host must have **rpc.ioadmd**(8C) running on it.

Enter the name of the desired host and do one of two actions: press the return key or select **System Map** from the menu associated with the **View** button.

Both the **View** and the **Do** buttons have an associated menu of selectable items. When an item is grayed-out it usually means that the action or information is not available for the selected device. A device must be selected before the item on the menu is available.

View button:

- Select **System Map** to get an overall view of the system. This display shows a server and each controller, disk, and tape associated with the system.

- Select **Information** to view general information about the currently selected device. Available for all devices except dedicated tape controllers.

- Select **Geometry** to view specific information relating to the physical layout of the disk. (Avaliable when a disk is selected.)

- Select **Partitioning** to get a better view of the defined partitions of a disk. (Avaliable when a disk is selected.)

- Select **Statistics** to view general I/O information about that disk. (Available when a disk is selected.)

- Select **Error Log** to view all levels of errors and messages for the selected device. Errors and

messages are filtered by **syslogd**(8).

• Select **Panic Log** when a disk has received panic errors. Panic errors on a disk are represented as an exclamation point (!) over the icon represented by the failing disk.

Do button:

• Select **Logging Status** to initiate error logging for **dmt**. This brings up a popup which displays information about **syslogd**(8) on the host. Pressing the left mouse button over **Start** automatically stops the currently running syslogd and starts a new one. The new one uses a **syslog.conf**(5) file created specifically for **dmt**.

• Select **Disk Test** to see if a disk is accessible through the **read**(2V) and **write**(2V) system calls.

• Select **File Test** to see if a file system on the disk is accessible through the **open**(2V), **close**(2), **read**(2V), and **write**(2V) system calls.

FILES

/var/spool/ioadm/dmtlog/alert.log	File of second most severe system errors which need immediate attention.
/var/spool/ioadm/dmtlog/crit.log	File of critical conditions that are given as warnings.
/var/spool/ioadm/dmtlog/emerg.log	File of panic conditions which need immediate attention. Will cause flag to be set on disk icon.
/var/spool/ioadm/dmtlog/err.log	File of less critical errors.
/var/spool/ioadm/dmtlog/info.log	File of all system messages.
/var/spool/ioadm/dmtlog/notice.log	File of messages which may require special handling (not critical).
/var/spool/ioadm/dmtlog/warning.log	File of warning messages (not critical)
/var/spool/ioadm/dmtlog/emerg.ar	File which contains size of emerg.log file the last time an update was performed.
/var/spool/ioadm/dmtlog/*device-name***.scrpad**	Scratch pad for the indicated device.
/var/spool/ioadm/dmtlog/syslog.conf	Configuration file for **syslogd**(8) taylored for **dmt**.

SEE ALSO

pmt(1), **fsm**(1), **getdomainname**(2), **gethostid**(2), **gethostent**(3N), **getpwent**(3), **getservent**(3N), **openprom**(4S), **syslog.conf**(5), **eeprom**(8S), **ioadmd**(8C), **dkinfo**(8), **syslogd**(8)

SunView 1 Beginner's Guide

SPARCserver Manager User's Guide

NAME

fsm – IOadmin File System Management (SPARCserver Manager)

SYNOPSIS

fsm [**–m**] [**–h** *hostname* [*hostname*]] [**–t** *seconds*]

AVAILABILITY

This tool is only available with the IOadmin tools which are part of the SPARCserver Manager software package.

DESCRIPTION

fsm is a SunView tool which, for a specified host, allows the user to:

- View disk and partition layout, including MetaDisk, in a graphical format.
- View information about files and file systems in a graphical format
- Create file systems, check file system integrity, mount/unmount file systems
- Activate and deactivate MetaDisk metapartitions and synchronize MetaDisk metamirrors

The data available through **fsm** is similar to that available through **dkinfo**, **df**, **du**, and various options of **ls**, but it is displayed in graphic fashion.

OPTIONS

–m Run in monochrome. Default is to run in color on color systems, monochrome on monochrome systems.

–h *hostname*

Come up ready to access *hostname*. The default is the local system. More than one *hostname* may be specified. A menu of names will be created from which to choose a server. Once the tool is started the user may change the host name at will.

–t *seconds*

Update the screen display every *seconds*. The default is 300 (5 minutes). To disable periodic update use a value of zero (e.g. fsm -t 0).

USAGE

For **fsm** to access a host, the host must have **rpc.ioadmd**(8C) running on it.

Enter the name of the desired host and do one of two actions: press the return key or select a display from the **View** button.

Both the **View** and the **Do** buttons have an associated menu of selectable items. The menu is displayed by moving the mouse pointer over the desired button, pressing and holding down the right mouse button. When an item is grayed-out it usually means that the action or information to be presented is related to a specific partition. The partition must be selected before the item on the menu is available. To select a partition move the mouse pointer over the desired partition and press the left mouse button.

View button:

- Select **Disk Layout** to get an overall view of the system. This display shows each disk associated with the system, including any MetaDisks. The disk displays are scaled relative to size as is the partitioning within each disk. The raw capacity of each disk is listed to the left of each display.

- Select **Partitions** to get a better view of how each partition is laid out relative to the other. Overlapping partitions can be viewed through this display. The raw capacity of each partition is listed to left of the display. In addition to viewing **All** the partitions of a specified system, the pull-right menu allows one to view only partitions **With file systems**, only partitions **Without file systems**, or only those partitions considered **Available**. Available can be taken to mean a partition which does not contain a mounted file system, is not used for swap, and is not a component of a meta partition.

- Select **Partition Detail** to get detailed information about the selected partition. This includes starting cylinder number, total number of cylinders, and raw capacity. If the selected partition contains a mounted file system information about the file system is displayed as well. If the partition is a meta

partition the layout of the physical partitions making up the meta partition are also displayed.

• Select **File System Usage** to get a report on how much disk space is occupied by each of the currently mounted file systems.

• Select **File Information** to get a information about a specific file or directory. This includes the percentage of the file system allocated to the specified file or directory tree.

Do button:

• Select **Update Display** to force an update of the screen display. For instance, if **File System Usage** is being displayed and a large file has recently been copied into one of the file systems the display will not reflect the copy until the next periodic screen update (default is every 5 minutes. Also see -t option). Use **Update Display** to note the change right away.

• Select **Create File System** to define a new file system on the selected partition.

• Select **Check File System** to check the integrity of the file system on the selected partition.

• Select **Mount File System** to mount the selected partition on the specified directory.

• Select **Unmount File System** to unmount the selected partition.

• Select Meta Partition and slide the cursor to the right to reveal Meta Partition options.

• Select **Activate** to bring a meta partition online.

• Select **Clear** to take a meta partition offline.

• Select **Resync** to synchronize a meta partition with its mirror partition.

See the *SPARCserver Manager User's Guide* for more information on MetaDisk.

SEE ALSO

pmt(1), **dmt**(1), **du**(1), **df**(1), **ls**(1), **fstab**(5), **mtab**(5), **metatab**(5), **dkinfo**(8), **newfs**(8), **fsck**(8), **mount**(8), **umount**(8), **metainit**(8), **ioadmd**(8C)

SunView 1 Beginner's Guide

SPARCserver Manager User's Guide

BUGS

The number of Unix file systems (UFS) on the File information display will not necessarily be correct for remote hosts.

Very small filesystems, such as those on floppy disks, are not displayed correctly and cannot be easily selected.

Commands from fsm (for example newfs(8)) directed at floppy disk partitions do not work correctly.

NAME

rpc.ioadmd – IOadmin RPC service (SPARCserver Manager)

SYNOPSIS

rpc.ioadmd

AVAILABILITY

This program is only available with the IOadmin tools which are part of the SPARCserver Manager software package.

DESCRIPTION

ioadmd is an RPC service which runs on a host and allows the IOadmin tools (running on a remote workstation) to monitor and manage some of the host resources.

USAGE

You must be super-user to start **rpc.ioadmd.**

FILES

/var/spool/ioadm/ioadmd.lock	Lock file to prevent more than one **rpc.ioadmd** from running at a time. This file should never need to be removed.

SEE ALSO

fsm(1), **dmt**(1), **pmt**(1), **pmlogd**(8)

SPARCserver Manager User's Guide

NAME

metatab – MetaDisk configuration database

SYNOPSIS

/etc/metatab

DESCRIPTION

/etc/metatab is the database used by the **MetaDisk.** Each MetaDisk metapartition is a minor device of the MetaDisk device, and is represented by one line in **/etc/metatab.** Each metapartition is either a concatenation of up to four real disk partitions, or a **metamirror** comprised of one or two regular metapartitions.

Metapartitions specified in **/etc/metatab** are activated or set up by the **/usr/etc/metainit(8)** command.

Each line in **/etc/metatab** is of the form:

special rows 1 real1 1 real2 ... 1 realn

which defines a disk concatenation of real partitions from real1 to realn, with its size set to be the sum of the sizes of all the real partitions,

or

special -m special options

which defines a metamirror that consists of a single mirrored metapartition, **special,** with its size set to be equal to the size of the mirrored metapartition,

or

special -m special1 special2 options

which defines a metamirror that consists of two mirrored metapartitions, **special1** and **special2,** with its size set to be the smaller of the two mirrored metapartitions.

Each of the mirrored metapartitions have to be defined separately in /etc/metatab like other metapartitions.

The fields have the following meaning:

special is the name of a metapartition, for example /dev/md0a, /dev/md2b . . ., etc. A total of 24 metapartitions can be configured and used. They are **/dev/md[0-2][a-h].**

rows is the number of rows in the metapartition. Each metapartition may consist of a certain number of rows of real partitions. The maximum allowable number of rows is four.

real is the name of the real disk partition. **/usr/etc/metainit** will complain if a real disk partition appears in more than one row of a metapartition, or in more than one metapartition. It will also complain if **rows** are not followed by a sufficient number of real partition names.

The pattern **1 real** is repeated up to four times, once for each row in the metapartition.

options:

The following flags control the mode of operation for disk mirroring:

-s means dispatch writes to the second mirrored metapartition (if it exists) after writes to the first mirrored metapartition has completed. Without this flag, all writes to the second mirrored metapartition will be dispatched in parallel with corresponding writes to the first mirrored metapartition.

-g means that when doing a read, if the disk block to be read is on the first half of the metamirror, read it from the first mirrored metapartition. Otherwise read it from the second mirrored metapartition.

-r means read only from the first mirrored metapartition. This flag should not be used with the -g flag. When neither flag is specified, the metamirror will alternate reads between the first and the second metapartitions.

-e number
specifies an error threshold for read errors before taking a faulty mirrored metapartition off-line. Note that a mirrored metapartition will be taken offline as soon as the first write error occurs.

single_digit
May be specified as the last entry of the option fields. It designates the pass number used for mirror resync during a system reboot. Mirror resync for mirrored pairs with smaller pass numbers will be scheduled before those with larger pass numbers. Mirror resync for metamirrors with equal pass number will be run concurrently. A **0** (zero) means to skip the mirror-resync during reboot. If the pass number is not specified, it defaults to **1.**

For further details, see examples below.

System V File-Creation Semantics
None

EXAMPLES

All IPI drives in the following examples have the same size of 2006130 sectors.

Concatenation:

This example shows a metapartition, /dev/md0c, consisting of a concatenation of two disks.

```
/dev/md0c 2 1 /dev/id000c 1 /dev/id001c
```

Since the first disk sector in /dev/id001c contains a disk label, it will be skipped by the MetaDisk. So /dev/md0c will contain 2006130*2-1 blocks, which is the total number of blocks on the two IPI disks, minus one. The first disk block from /dev/id001c is not used by /dev/md0c. The second disk block on /dev/id001c corresponds to block 2006130 on /dev/md0c.

Mirroring:

This example shows a configuration of a metamirror, /dev/md2a, comprised of two metapartitions, /dev/md0a and /dev/md0b.

```
/dev/md2a -m /dev/md0a /dev/md0b -e 10 2
/dev/md0a 1 1 /dev/id001c
/dev/md0b 1 1 /dev/id010c
```

Any write request to metapartition /dev/md2a will be written to both /dev/md0a and /dev/md0b; data will be duplicated on /dev/id001c and /dev/id010c.

Even numbered blocks will be read from /dev/id001c. Odd numbered blocks will be read from /dev/id010c. Therefore, the IO load for the reads will be evenly distributed between the two disks.

You should be aware of some performance implications when using disk mirroring: to minimize the performance penalty of disk mirroring, you should put all mirrored metapartitions on different controllers and you should not use the -s flag. Then the writes will be dispatched to the mirrored metapartitions in parallel, and the elapsed time for completing a write will be reduced.

Concatenation+Mirroring:

In this example, two metapartitions are defined, each consisting of a concatenation of two IPI disks. Then a metamirror is defined on top of the two metapartitions.

```
/dev/md1c 2 1 /dev/id002c 1 /dev/id003c
/dev/md1d 2 1 /dev/id010c 1 /dev/id011c
/dev/md2c -m /dev/md1c /dev/md1d -g -s
```

The size of /dev/md2c is 4012259 blocks, the same as /dev/md1c and /dev/md1d.

The -s flag means that writes to /dev/md1d will only be done after writes to /dev/md1c are completed. This means that /dev/md1c may be more **up-to-date** than /dev/md1d at any given time.

The **-g** flag means that blocks are read only from /dev/id002c and /dev/id011c.

Logic indicates that if one of the real partitions has a disk failure, you should replace the failing disk as soon as possible, and run the Metadisk utility **/usr/etc/metasync,** which allows resyncing of the mirrored pair with minimal impact on the availability of the metapartition.

FILES

/etc/metatab

SEE ALSO

fstab(5), **metautil**(8), **SPARCserver Manager User Guide.**

BUGS

Recursive mirroring is not allowed; that is, a metamirror cannot be a component in the definition of another metamirror.

NAME

MetaDisk utilities

SYNOPSIS

/usr/etc/metainit -a|special
/usr/etc/metainit -r
/usr/etc/metaclear -a|special
/usr/etc/metastat [-p] [special]
/usr/etc/metasync [buffer_size] metamirror special-from
/usr/etc/metasync -r
/usr/etc/metattach metamirror special
/usr/etc/metadetach metamirror special
/usr/etc/metareplace metamirror special-from special-to
/usr/etc/metaoffline special
/usr/etc/metareset special
/usr/etc/metad
/usr/etc/metahalt
/usr/etc/metafasthalt
/usr/etc/metareboot
/usr/etc/metafastboot

DESCRIPTION

/usr/etc/metainit

sets up or activates metapartitions according to the configuration specified in **/etc/metatab;** You must activate a metapartition before you can use it. The metainit command reads a succession of lines from the file **/etc/metatab,** each line representing one metapartition of the MetaDisk. If followed by **-a,** it activates all the metapartitions defined in /etc/metatab. Otherwise it only activates the metapartition specified.

/usr/etc/metainit -r

is used only during a system reboot. It sets up or activates all metapartitions that were active before the system was shutdown or crashed. It should be added in **rc.boot** before fsck so that metapartitions will be activated automatically during a reboot.

/usr/etc/metaclear -a

clears or deactivates all active metapartitions. If followed by **special** instead of -a, it will only clear the specified metapartition. After a metapartition is cleared, it can no longer be used.

/usr/etc/metastat

displays the current status for each metapartition that is active. If followed by a **special,** it will only display the status for the specified metapartition.

-p option means to print the list of active metapartitions in **metatab(5)** format that can be used in /etc/metatab.

/usr/etc/metasync [buffer_size] metamirror special-from

will start a mirror resync operation from **special-from** under **metamirror** to the other mirrored metapartition under the same metamirror.

buffer_size gives the size (number of disk blocks) of the internal copy buffer for the mirror resync. The size is defaulted to 64 disk blocks (32K). It can be no greater than 126.

/usr/etc/metasync -r

handles special mirror resync requirement during a system reboot. It should only be run from **rc.local.** It will schedule all the mirror resync jobs according to their pass numbers. It will only do mirror resync on those metamirrors where a mirror resync is necessary.

/usr/etc/metattach metamirror special

attaches a metapartition **special** to a **metamirror.** Since a metamirror can have only one or two mirrored metapartitions, you can only attach one more mirrored metapartition to a metamirror which has already has one mirrored metapartition.

After a successful metattach, the newly added metapartition will usually not be in sync with the other mirrored metapartition. It will be attached in an **off-lined** state. No reads or writes will be directed to it. Only after you have successfully run metasync to completion will the new metapartition be brought on-line and used.

/usr/etc/metadetach metamirror special

will detach a mirrored metapartition **special** from its **metamirror.** After a metadetach, the metapartition **special** becomes a stand-alone metapartition by itself.

If **special** is the only mirrored metapartition left in the metamirror, it can't be detached.

Because the state of the file system on the metapartition is inconsistent after the metadetach (there may be dirty pages in the buffer cache that have not yet been flushed to the disk), you must run fsck to ensure that the file system is intact; otherwise the system may panic and crash. If you do not have a file system mounted on the metapartition, use whatever partition integrity checks are appropriate for the application.

/usr/etc/metareplace metamirror special-from special-to

replaces a real disk partition **special-from** under **metamirror** with **special-to.** After the replacement, the mirrored metapartition where the new disk partition is attaching to will be put **off-line** by MetaDisk because it will be out-of-sync with the other mirrored metapartition. You have to run **metasync** to get it in sync with the other mirrored metapartition and get it back on-line.

Note that the the new disk partition has to be large enough to replace the bad disk partition. Note also that metareplace modifies /etc/metatab, substituting the new physical partition for the one that was replaced.

/usr/etc/metaoffline special

off-lines the mirrored metapartition **special** from a metamirror. After metaoffline, the MetaDisk will stop reading/writing to the off-lined mirrored metapartition.

Note that **metaoffline** does not sever the logical association between the mirrored metapartition and the metamirror, as does **metadetach.** If you want to take a mirrored metapartition completely away from a metamirror, use metadetach.

/usr/etc/metareset special

resets the "offlined" flag for a mirrored metapartition. The effect is that the MetaDisk will start reading/writing to the mirrored metapartition.

Note that this command has to be used with a great caution. It does not try to repair a bad metapartition, nor does it try to bring an out-of-sync mirror metapartition back in sync with the other mirrored metapartition. It simply **forces** the MetaDisk to put the mirrored metapartition back on-line and to start using it again.

If you bring an out-of-sync metapartition online with metareset, there is a good chance you will panic your system. Do not bring an off-lined mirrored metapartition back on-line, therefore, unless:

a) you are sure that the off-lined mirrored metapartition is in-sync with the other mirrored metapartition,

or

b) your application on top of the metamirror knows how to handle it,

or

c) you don't need the mirrored pair to be in sync to start with (for example, if you want to start from scratch and run newfs to put an empty file system on it).

Otherwise, you should run **/usr/etc/metasync** to bring the out-of-sync mirrored metapartition back in-sync with the other.

/usr/etc/metad

This is the MetaDisk daemon. It is created automatically the first time **/usr/etc/metainit** is executed. Its purpose is to handle error conditions and automatic failover for metamirrors.

/usr/etc/metahalt
/usr/etc/metafasthalt
/usr/etc/metareboot
/usr/etc/metafastboot

are similar to

/usr/etc/halt
/usr/etc/fasthalt
/usr/etc/reboot
/usr/etc/fastboot

except that when using the former commands, mirror resync during the subsequent reboot will be skipped.

DESCRIPTION

System V File-Creation Semantics

None

EXAMPLES

metainit, metastat:

```
lima# cat /etc/metatab
/dev/md1a -m /dev/md0a /dev/md0b -g
/dev/md0a 1 1 /dev/id001c
/dev/md0b 1 1 /dev/id012c


lima# metainit -a
special /dev/md0a is set up
/dev/md0a:
     1 rows, total blocks 2006130
     row 0:  width 1 blocks 2006130  devices:      < /dev/id001c >

special /dev/md0b is set up
```

```
/dev/md0b:
     1 rows, total blocks 2006130
     row 0:  width 1 blocks 2006130  devices:      < /dev/id012c >

special /dev/md1a is set up
/dev/md1a:
                          devices:      < /dev/md0a >
                          devices:      < /dev/md0b >

lima# metastat
*****
/dev/md1a: metamirror
     first mirror partition is /dev/md0a
     second mirror partition is /dev/md0b
     error threshold = 0   pass = 1
     read the first half from the first mirror partition, second half
          from the second mirror partition
     dispatch writes to the first and second mirror partition in parallel
  /dev/md0a: (first)
     1 rows, total blocks 2006130
     row 0:  width 1 blocks 2006130  devices:      < /dev/id001c >

  /dev/md0b: (second)
     1 rows, total blocks 2006130
     row 0:  width 1 blocks 2006130  devices:      < /dev/id012c >

*****
```

FILES

/etc/metatab

SEE ALSO

metatab(5) **mount**(2), **unmount**(2), **fstab**(5),

BUGS

NAME

pmlogd – IOadmin performance data logger (SPARCserver Manager)

SYNOPSIS

pmlogd [*sample-interval*]

AVAILABILITY

This program is only available with the IOadmin tools whaich are part of the SPARCserver Manager software package.

DESCRIPTION

pmlogd runs on a host and periodically samples kernel performance data using **rstat**(3R). **pmlogd** logs the kernel data to a disk file in ASCII format for use by the **pmt** IOadmin tool.

OPTIONS

sample-interval

Sample and log data every *sample-interval* minutes. Default is 20.

USAGE

You must be super-user to start **pmlogd**, and only one **pmlogd** may be running at a time.

pmlogd writes data to **/var/spool/ioadm/pmlog**; one file per day. The file names are of the format **pm***yymmdd*. **pmt**(1) uses **pmlogd** to record data for future viewing.

FILES

/var/spool/ioadm/pmlog/pm*yymmdd*

ASCII log files containing performance data; one file for each day.

/var/spool/ioadm/pmlog/pmlogd.lock

Lock file to prevent more than one **pmlogd** from running at a time. This file should never need to be removed.

SEE ALSO

pmt(1), **rstat**(3R)

SPARCserver Manager User's Guide

WARNINGS

Logging with a short sample interval over a long period can consume significant amounts of disk space.

NAME

pmt – IOadmin performance monitor (SPARCserver Manager)

SYNOPSIS

pmt [**–m**] [**–h** *hostname* [*hostname*]]

AVAILABILITY

This tool is only available with the IOadmin tools which are part of the SPARCserver Manager software package.

DESCRIPTION

pmt is a SunView tool which, for a specified host, allows the user to:

- Control the logging of performance data to disk
- View logged performance data in a graphical format
- View current performance data in a graphical format

The data available through **pmt** is the same as that available through **perfmeter**, but it is displayed in greater detail.

OPTIONS

–m Run in monochrome. Default is to run in color on color systems, monochrome on monochrome systems.

–h *hostname*

Come up ready to access *hostname*. Default is the local system. More than one *hostname* may be specified. A menu of names will be created from which to choose a server. Once in the tool the user may change the host name at will.

USAGE

For **pmt** to access a host, the host must have **rpc.ioadmd**(8C) running on it, and **rpc.rstatd**(8C) available on it.

To initiate logging on the specified host select **Logging Status** from the **Do** button menu. This brings up a popup which displays information about the log daemon on the host, and allows you to stop and start data logging (see **pmlogd**(8)).

To view data which has been logged to disk on the host, select **Logged Data** from the **View** button menu. This brings up a popup which allows you to load data for a particular time period.

To view data in realtime, select **Current Data** from the **View** button menu.

FILES

/var/spool/ioadm/pmlog/pm*yymmdd*	Log files containing historical data, one file for each day.
/var/spool/ioadm/pmlog/pmlogd.lock	Lock file to prevent more than one **pmlogd(8)** from running at a time. You should never have to remove this file on re-starts.

SEE ALSO

fsm(1), **dmt**(1), **perfmeter**(1), **ioadmd**(8C), **pmlogd**(8), **rstatd**(8C)

SunView 1 Beginner's Guide

SPARCserver Manager User's Guide

WARNINGS

Logging with a short sample interval over a long period can consume significant amounts of disk space.

BUGS

Index

A

a partition, 30
activating all metapartitions at once, 33
activating metamirror, 36
activating metapartitions, 32, 39, 49
activating metapartitions during reboot, 19
activating one metapartition, 34
adding second metapartition to metamirror, 50
attaching a metapartition, 25
automatic metadisk daemon startups, 19
automatic metapartition activation, 19
automatic metapartition mounting, 21
automatic restart, dmt logging, 14
automatic resync, 20

B

b partition, 30
backups, mirroring active, 55
backups, no mirroring, 57
basics, metadisk driver, 23
block or raw device names, 41
booting new kernel, 17
building MetaDisk Driver kernel, 15
buttons and menus, 70
buttons, cycle, 72
buttons, mouse, 70

C

c partition, 30
cautions, metadisk, 40
changing servers, 77
checking metapartition status, 49
clearing metapartitions, 49
clearing panic error log, 87
closing dmt to an icon, 82
closing tool to icon, 79
command line options, 73
command path setting, IOadmin, 14
commands, metadisk, 25
concatenated metamirror, 47
concatenation definition, 25
customized kernel, 17
cycle buttons, 72

D

daemon startups, automatic, 13
daemon, rpc start, 12
defining a metamirror, 46
defining metapartitions, 30, 39, 45
definition of terms, 2
definitions, metadisk, 24
detaching a metapartition, 25
Device Management, 81
Device Management window, 82
device names, raw, 30
directories, IOadmin, 67
disjoint reads, metamirror, 47
disk space requirements, 7, 10
disk test, 89
disk writes, not to first block, 41
dismissing popup windows, 69
display update, 88
dmt error log, 86
dmt logging, 88
dmt logging, automatic restart, 14
dmt system map, 82
do and view menus, 71
do menu, dmt, 87
do menus, 68
document organization, 1
dynamic backup, 25

E

error conditions, metadisk, 34
error handling, metadisk, 64
error log, clearing panic, 87
error log, displaying, 87
error log, dmt, 86

F

failed disk partition, replacing, 61, 62
failed disk, replacing, 60
file system, making & mounting, 53
file systems on metapartitions, 39
file test, 90
files affected by MetaDisk Driver installation, 15
first block of disk, no writes, 41
fsm, speeding up, 75

fsm, using with MetaDisk Driver, 32

G

generic kernel, 16
getting started, 5

H

hardware & software requirements, 5
hardware and software requirements, 6
hot spare, 25

I

icon, closing to, 79
icon, opening from, 79
icons, closing & opening, 79
information popups, 85
installation, local, 6
installation, procedures after, 12
installation, remote, 9
installation, space requirements, 7, 10
installing software, 39
introduction, 1
ioadm daemon startups, automatic, 13
ioadm tools, mounting, 15
ioadmd.conf, 76
ioadmd.conf contents, 74
ioadmd.conf file, 74
ioadmd.conf, preventing side effects, 74
ioadmd.conf, sample file, 77
IOadmin directories, 67
IOadmin requirements, 6
IOadmin tool windows, 68
IOadmin tools, quitting, 79
IOadmin tools, starting, 73
IOadmin, starting, 72

K

kernel booting, 16, 17, 18
kernel build, 39
kernel building, 17
kernel naming, 17, 18
kernel, building MetaDisk Driver, 15
kernel, customized, 17
kernel, generic, 16
kernel, making, 18
kernel, making new, 16

L

local installation, 6
log files, 80
logging, dmt, 14

M

making custom kernel, 17
making generic kernel, 16
making new kernel, 16
making physical partition into metapartition, 51
man pages, 19
man pages, moving, 14
menu, do, 87
menu, server name, 71
menu, view, 84
menus and buttons, 70
menus, view and do, 71
metaclear, 26
metadetach, 27
metadisk cautions, 40
metadisk commands, 25
metadisk commands in root's path, 19
metadisk daemon, 26
metadisk daemon startups, automatic, 19
metadisk definitions, 24
metadisk driver basics, 23
MetaDisk Driver installation, files affected, 15
MetaDisk Driver kernel building, 15
MetaDisk Driver problems, 32
metadisk driver requirements, 5
Metadisk Driver tutorial, 39
metadisk error conditions, 34
metadisk error handling, 64
metadisk overview, 24
metadisk performance, 29
metadisk performance tips, 65
metadisk post-installation, 18
metadisk reliability, 29
metadisk safety tips, 63
metadisk, customized kernel, 17
metadisk, generic kernel, 16
metadisk, summary instructions, 39
metadisk, when to use, 24
metafastboot, 28
metafasthalt, 28
metahalt, 28
metainit, 26, 32
metamirror, 47
metamirror definition, 24
metamirror, -g option, 47
metamirror, -r option, 47
metamirror, -s option, 47
metamirror, adding second metapartition, 50
metamirror, concatenated, 47
metamirror, defining, 46
metamirror, one metapartition defined, 46
metamirror, read-error threshold, 48
metaoffline, 27
metapartition definition, 24, 45
metapartition naming, 29
metapartition planning, 28
metapartition status checking, 49
metapartition, modifying, 50
metapartition, modifying definition, 59
metapartition, offlined, 25
metapartition, status checking, 36
metapartitions, activating, 32, 49
metapartitions, clearing, 49
metapartitions, file systems mounted, 39

metapartitions, use of, 53
metareboot, 28
metareplace, 27
metareset, 27
metastat, 26
metastat command, 36
metasync, 27
metatab definition, 24
metatab entry from metastat, 46
metatab file, 30
metatab file sample, 31
metatab sample, 48
metattach, 27, 51
mirrored metapartitions, syncing, 54
mirroring definition, 25
modifying a used metapartition, 59
modifying metapartition definition, 59
modifying unused metapartition, 50
mounting a file system, 53
mounting during reboot, 21
mounting IOadmin tools, 15
mouse buttons, 70
moving a window , 70

N

naming metapartitions, 29
newfs command, 53
note pad, 86

O

offlined metapartition, 25
online backup, 25
online backups, mirroring active, 55
online backups, no mirroring, 57
opening tool from icon, 79
options, command line, 73
organization of document, 1
overview, vii
overview, metadisk, 24
overview, metadisk driver, 23

P

partitions a, b, c, 30
partitions with file systems, 98
path setting, metadisk, 19
path, command setting (IOadmin), 14
performance tips, metadisk, 65
performance, metadisk, 29
planning metapartitions, 28
popup windows, 69, 85
post-installation procedures, 12
post-installation, metadisk, 18
printing screen, 70

Q

quitting dmt, 82
quitting IOadmin tools, 79

R

raw device names, 30
raw or block device names, 41
read-error threshold, metamirror, 48
read-from-first option, 47
reboot without resyncing, 63
reboot, activating metapartitions, 19
reboot, automatic resync, 20
rebooting new kernel, 16, 18
reliability, metadisk, 29
remote installation, 9
replacing failed disk, 60
replacing failed disk partition, 61, 62
requirements, hardware & software, 5
requirements, IOadmin, 6
requirements, metadisk driver, 5
resizing a window , 70
resync after reboot, 20
resync, watching progress, 55
root password needed, 71
rpc daemon, start, 12

S

safety tips, metadisk, 63
sample metatab, 48
sample metatab file, 31
screen printing, 70
scrollable canvases and windows , 69
serial-read option, 47
server name menu, 71
servers to be monitored, 73
servers, changing, 77
simple metapartition, 45
software requirements, 6
space requirements, installation, 7, 10
SPARCserver Manager overview, vii
spare disk, 25
speeding up fsm, 75
starting Device Management (dmt), 81
starting dmt logging, 88
starting File System Management (fsm), 94
starting IOadmin tools, 72, 73
starting Performance Monitor (pmt), 107
starting rpc daemon, 12
status checking, metapartition, 36
status indicators, dmt, 84
summary metadisk instructions, 39
SunView environment, 67
syncing definition, 25
syncing mirrored metapartitions, 54
system map, 83
system map, dmt, 82

T

terms, definitions of, 2
text subwindows, 69
tutorial, Metadisk Driver, 39
typographic conventions, 2

U

updating display, 88
used metapartition, modifying, 59
using metapartitions, 33, 53

V

view and do menus, 71
view menu, dmt, 84
view menus, 68
vmunix, 17, 18

W

when to use metadisk, 24
window, dmt, 82
window, moving, 70
window, resizing, 70
windows, 67
windows, popup, 85
windows, primary, 68